LA PRIMA RIVISTA EUROPEA DI SOFTWARE PER PERSONAL COMPUTER

ISSN 0392-8896

PERSONAL SOFTWARE

ANNO 2 N. 10/11
SETTEMBRE-OTTOBRE 1983 L. 3.500

UNA PUBBLICAZIONE DEL GRUPPO EDITORIALE JACKSON



Spedizione in abb. postale Gruppo III/70

INTRODUZIONE ALL'INTELLIGENZA ARTIFICIALE

AREA PER LINGUAGGIO MACCHINA NEL BASIC ZX 81

IL TAGLIO DEL TI99/4A

GIOCHI DI INSEGUIMENTO

DEDALO 3-D

CONVERSIONI DI PROGRAMMI

Usare il sistema operativo CP/M

IL LIBRO

Il sistema operativo CP/M è stato progettato per rendere semplice l'uso di un microcomputer. Questo libro vi renderà semplice l'uso del CP/M. (Le versioni esaminate del CP/M sono il CP/M 1.4-il CP/M 2.2. e il nuovo sistema operativo multiutente MP/M) La maggior parte di utenti di microcomputer dovrà, infatti, un giorno o l'altro, fare ricorso al CP/M, disponibile su quasi tutti i computer basati sui microprocessori 8080 e Z80, come pure su certi sistemi utilizzanti il 6502. Il libro, senza presupporre alcuna conoscenza di un calcolatore, inizia con la descrizione, passo-passo delle procedure di inizializzazione del sistema: accensione, inserimento dei dischetti, esecuzione delle più comuni operazioni su file, compresa la duplicazione dei dischetti. Prosegue con il PIP (programma di trasferimento dei file), il DDT (programma di messa a punto) e ED (programma editor). Per entrare sempre più, fornendo numerosi consigli pratici, all'interno del CP/M e delle sue operazioni, al fine di comprenderne appieno le risorse ed eventualmente dare gli strumenti per successive modifiche.

SOMMARIO

Introduzione al CP/M e all'MP/M-Le caratteristiche del CP/M e dell'MP/M-Gestione dei file con PIP-L'uso dell'editor-Dentro al CP/M e all'MP/M-Guida di riferimento ai comandi e ai programmi del CP/M e dell'MP/M-Consigli pratici-Il futuro-messaggi comuni di errore-tabella di controllo di ED-nomi dei dispositivi di PIP-riassunti dei comandi-parole chiave di PIP-parametri di PIP-tasti di controllo per la digitazione dei comandi-tipi di estensione-lista dei materiali-organizzazione della stanza del calcolatore-verifiche in caso di errore-regole di base per la localizzazione dei guasti.

Pagg. 320 Cod. 510P
L.22.000 (Abb. L.19.800)

Per ordinare il volume utilizzare l'apposito tagliando inserito in fondo alla rivista.



GRUPPO EDITORIALE JACKSON
Divisione Libri

un nuovo corso per imparare a dialogare con il personal computer



Il personal computer: un protagonista

Il computer è figlio dell'informatica, la scienza degli anni '80, che sta rivoluzionando il mondo della produzione e, in un futuro non molto remoto, trasformerà radicalmente la qualità della nostra vita.
I computers sono ormai pronti a lavorare per noi: ora siamo noi che dobbiamo imparare a comunicare con loro, per metterli in grado, con le nostre istruzioni, di fornirci il maggior numero di prestazioni e al piú elevato livello.
È in quest'ottica che INFOR ha messo a punto il suo corso di informatica di base programmato per l'insegnamento a distanza, che rappresenta lo strumento più perfezionato oggi reperibile per chi vuole trattare da pari a pari con il proprio elaboratore.

Un corso per tutti

Questo corso teorico-pratico è indispensabile per chiunque, già inserito nel mondo o nel mercato del lavoro, desideri accostarsi all'informatica per migliorare le proprie capacità produttive. E anche per i giovani che vogliono inserirsi in questo nuovo mondo.

La gestione del-computer; il linguaggio BASIC

Il corso è facilmente comprensibile a chi si avvicina per la prima volta all'informatica: ne insegna l'abc con un linguaggio semplice e piano, e mettendo fin dalla prima lezione l'utente a contatto diretto con il personal, in poche settimane e gli insegna, comodamente a casa, a farlo funzionare a programmarlo in BASIC. A fine corso INFOR rilascia un attestato a conferma della preparazione raggiunta.

Altre 7 proposte INFOR: Giornalista, Fotografo, Interprete, Grafico pubblicitario, Tecnico pubblicitario, Programmista radio-TV, Audiovisivi.



ISTITUTO SUPERIORE PER LA COMUNICAZIONE E L'INFORMAZIONE

INFORMAZIONI URGENTI: TEL. 06/62.30.341

INFOR Via G.V. Englen, 25/T 00163 Roma

Desidero ricevere informazioni sul vostro corso ______

Cognome ______ Nome ______ Età ______

Professione ______ Via ______ N. ______

Città ______ C.A.P. ______ Prov. ______

Motivo della richiesta ☐ studio ☐ lavoro ☐ hobby

T 0 2 0

Se risiedi in Svizzera, scrivi direttamente a: IBERCULTURA GmbH, Baselstrasse 67 - 6003 Luzern - Tel. 041/226.617

ANNO 2 N.10-11 SETTEMBRE-OTTOBRE 1983

PERSONAL SOFTWARE



In copertina: il personal in cattedra. Una rappresentazione fantasiosa per un numero dedicato all'intelligenza artificiale.

ARTICOLI

RUBRICHE

GUIDA

In questa guida sono riportati i personal computer e i microprocessori di cui si parla negli articoli e nelle rubriche.

INIZIARE NEL MODO MIGLIORE

Guida al SINCLAIR ZX81 ZX80-Nuova ROM



Pagg. 262
Cod. 318B

L. 16.500
(Abb. L. 14.850)

IL LIBRO
Questa guida, con chiarezza, semplicità espositiva e ricchezza di esemplificazioni, risulta un vero e proprio strumento operativo per tutti coloro che vogliono avvicinarsi all'informatica in generale, e imparare la programmazione in BASIC, in particolare travalicando i tre calcolatori (ZX81, ZX80, ZX80 nuova ROM) a cui fa riferimento. Partendo da quello che è un computer, il lettore impara nei primi sei capitoli a programmare in BASIC, spingendosi, per chi lo vuole, oltre, sino alla programmazione in linguaggio macchina. L'ultimo capitolo riporta parecchi programmi e per ciascuno, vengono fornite, dove possibile, le diverse versioni. Tra l'altro si parlerà di file e di animazione delle figure. Per finire ben otto Appendici, essenziali ed utilissime, tra cui spiccano per interesse le due dedicate ai sistemi operativi dello ZX80, ZX80 nuova ROM e ZX81.

SOMMARIO
Introduzione - Il calcolatore - Installazione del calcolatore - La programmazione - Il linguaggio BASIC - Come operare - Utilizzo della memoria - Linguaggio macchina - Esempi di programmi ---- caratteri del sistema - variabili del sistema - scheda BASIC ZX80 - scheda BASIC ZX80 nuova ROM e ZX81 - errori segnalati dalla macchina - sistema operativo dello ZX80 - sistema operativo dello ZX81 e nuova ROM

Per ordinare il volume utilizzare l'apposito tagliando inserito in fondo alla rivista.



GRUPPO EDITORIALE JACKSON
Divisione Libri

Il Grimaldello elettronico

Riccardo Paolillo

Siamo abituati a non stupirci più per ciò che accade nel magico mondo dell'informatica, eppure confesso di essere rimasto alquanto perplesso leggendo poche settimane fa il resoconto, riportato con un certo rilievo da un importante quotidiano, dell'"impresa" riuscita ad alcuni ragazzi americani.

Si raccontava della vicenda di 12 ragazzi di Milwaukee, tutti tra i 15 e i 22 anni, che venuti a conoscenza chissà come degli opportuni codici di accesso, avevano utilizzato i loro personal per scorrazzare negli archivi magnetici di un laboratorio di ricerche sulle armi nucleari, di una banca e di una scuola, provocando, pare, anche alcuni danni.

La notizia, di per sé non sconvolgente anche perché il taglio dell'articolo lasciava intravedere la possibilità che un fatto certamente vero fosse stato esposto in maniera alquanto colorita a beneficio del grande pubblico, mi ha tuttavia spinto a qualche considerazione.

La situazione attuale della nostra telematica, non certo orientata ad un coinvolgimento a livello "personal", rende estremamente lontani da noi e poco probabili episodi come questo. Questa situazione, se da un lato penalizza gli utenti dell'informatica personale in quanto li priva della tanto sognata possibilità di collegarsi con il proprio personal a banche di software mediante un semplice accoppiatore acustico, dall'altro li costringe a realizzare i propri programmi e diventare quindi degli esperti nella affascinante arte della programmazione.

Quale è il futuro (prossimo) da augurarsi per l'informatica personale? Come tutte le cose, nella ricerca di uno sfruttamento equilibrato ed intelligente di tutte le possibilità che ci verranno offerte.

Quindi sì (eccome!) alla nascita e sviluppo di archivi software accessibili via rete telefonica, ma tenendo sempre presente che il nostro personal oltre a fungere da terminale per l'interscambio di programmi (soprattutto giochi) e da videogame per il loro utilizzo, sarà pur sempre un calcolatore sul quale sviluppare idee "personali" sempre nuove.

Con questo discorso, voglio ribadire un concetto per noi molto importante, che ci guida nella redazione di ogni numero della rivista. Accanto ad un certo numero di programmi pronti che appagano quanti vogliono "vedere subito qualcosa", cerchiamo di proporvi degli strumenti software (tool), sia programmi di utilità che informazioni descrittive, con lo scopo di stimolarvi a cercare nuovi campi di utilizzo per il vostro personal.

In questo numero troverete, a grande richiesta, parecchi programmi per le macchine più popolari, le rubriche sempre più interessanti, grazie ai vostri contributi, e numerosi articoli per tutti i gusti, tra cui vi segnalo la seconda puntata sulla conversione al BASIC dello ZX 81 di B. Del Medico e due articoli che sicuramente vi avvinceranno: *Il taglio del TI99/4A* di A. Chessa (ma è interessante per tutti) che vi introdurrà all'affascinante mondo dei giochi di strategia, e i programmi di simulazione di F. Sardo che vi illustrerà in modo sintetico ma rigoroso un utilizzo finora poco frequentato dei personal.



E ADESSO CHE ESPORTO CAPPELLINI GIALLI, CALZONCINI VERDI E MAGLIETTE ROSSE, CHI MI AIUTERA' A TINGERE DI ROSA IL FUTURO DELL'AZIENDA?



IL PERSONAL COMPUTER IBM IL TUO PICCOLO GRANDE AMICO.

GGK

Le statistiche di mercato possono fare molto per il futuro della tua azienda. A patto che siano precise, aggiornate e conformi alle tue esigenze. Devi poter leggere tra le righe, insomma. E con il video a colori del Personal Computer IBM potrai affrontare i tuoi problemi, vedendo sempre chiaro anche nei dati più oscuri. Ma non solo. Il Personal Computer IBM può aiutarti a fare di tutto perchè riceve dati, analizza, calcola, registra, stampa e, grazie alla sua potente memoria e ai minidischi, ti consente di archiviare un'infinità di informazioni. Ogni cosa sarà più semplice con un amico così. Vuoi metterlo alla prova?
Vai da un concessionario IBM. Scegli quello che ti è più comodo, nell'elenco della pagina che segue.



Il Personal Computer IBM contiene un microprocessore a 16 bit e una memoria di utilizzo che raggiunge i 640 Kbyte, e può essere dotato di un video a colori e di un coprocessore matematico. E, grazie ai dischi fissi, la capacità massima di memoria del sistema è di 21 Mbyte in linea. Inoltre, puoi facilmente collegarti con un altro Personal Computer IBM, con elaboratori più potenti e con la rete dei Centri Servizi Elaborazione Dati della IBM.

Sistemi operativi: DOS 1 - DOS 2 - UCSD - CP/M-86. **Supporti per le comunicazioni:** Asincrone - SDLC - BSC - Emulazione: 3101-3270. **Linguaggi:** tutti i principali e in più l'APL. **Programmi applicativi per:** aziende e servizi - produttività individuale - ufficio moderno - calcolo tecnico e scientifico - applicazioni professionali - didattica.

Richieste Spectrum

Spettabile Redazione,
sono un vostro lettore da Giugno, e cioè da quando ho comprato uno ZX Spectrum (con il quale mi trovo molto bene).
Vorrei a questo punto porvi delle domande:
- pubblicherete un servizio per imparare a programmare in linguaggio macchina sullo ZX Spectrum?
- Potreste cercare di dedicare uno spazio maggiore allo Spectrum anziché ad altri personal meno diffusi?
- Potreste pubblicare il programma "Collisione" (del numero 3) convertito per lo Spectrum?

Distinti saluti.

Riccardo Nicoletti
Firenze

Col prossimo numero, prenderà il via una serie di articoli dedicati proprio alla programmazione in linguaggio macchina. In tal modo speriamo di soddisfare anche le richieste di molti altri che ci hanno più volte invitato ad occuparci dell'argomento. Come è ormai nostra consuetudine, abbiamo scelto un approccio molto semplificato a beneficio dei meno esperti.

Per quanto riguarda la seconda richiesta, ne prendiamo nota, ma facciamo benevolmente osservare che sono tantissime le lettere che giungono in Redazione del tipo: perché al mio personal zy che è tanto diffuso dedicate così poco spazio, mentre pubblicate tanti programmi per yz che invece non compra nessuno? È chiaro che molto spesso le valutazioni sulle percentuali di diffusione dei personal "rivali" sono quantomeno discutibili e magari dettate da un po' di comprensibile "partigianeria". Inoltre occorre tener conto del fatto che i contributi che riceviamo non hanno tutti lo stesso livello qualitativo e di interesse. A questo proposito, rinnoviamo a tutti l'invito a collaborare, ma tenendo sempre presente gli indispensabili requisiti di originalità e interesse per un vasto pubblico, che ogni lavoro deve necessariamente avere per poter essere pubblicato. Per coloro che ritengono di avere qualche idea interessante, è disponibile l'ultima edizione della Guida agli autori.

Per quanto riguarda, infine, la conversione di "Collisione" rilanciamo la richiesta ai molti Sinclairisti.

Un simbolo misterioso

Egregia Redazione,
mi riferisco all'articolo "Programmare grafici con il TI99/4A" pubblicato nel n. 8-9. Orbene il passo 1290 è il seguente: 1290 H = H + VAL (SEG$ (A$, Z, 1)) * 2↑ (LEN (A$) – 2). Quella freccetta che cosa significa?

Sul mio TI non esiste, ho tentato con tutti i tasti, e con il tasto "function", ma la freccetta non è uscita fuori: è il mio computer guasto o è una bizzarria dell'autore? Ho notato comunque che la freccia ricompare anche dopo, quindi cosa devo fare? Grazie in anticipo.

Enrico Ferrari
Roma

Rassicuriamo subito il nostro lettore che il suo personal funziona perfettamente. In effetti la misteriosa freccetta altro non è che il simbolo di elevazione a potenza, operazione che nel TI viene eseguita mediante la digitazione del tasto ^ (SHIFT-6).

Quando il BASIC non è standard

Spett.le Redazione,
ho tentato di convertire il vostro "gioco del calcio" scritto per PET/CBM sul mio Spectrum. Dopo una faticaccia, peraltro stupida dovuta al fatto che sono un novizio, il programma non gira. E non poteva essere altrimenti non conoscendo cosa c'è nelle locazioni di memoria (quei maledetti CHR$ (...)) del PET.

Dal risultato, ovviamente, viene fuori che ci sono cose diverse.

Perché non create voi una rubrica in cui si riportano tutti i contenuti delle locazioni di memoria dei diversi home computer?

Da parte nostra la cosa è molto difficile, perché chi compra un home computer, non può spendere altri soldi per comperare altri personal solo per avere il manuale che gli spieghi (quelli che lo fanno) cosa c'è in memoria.

Matteo Mariani
Roma

Il problema della mancanza di compatibilità tra i vari personal costituisce uno dei maggiori motivi di malcontento, soprattutto per i principianti, e le osservazioni del Sig. Mariani sono infatti condivise in numerose lettere di altri lettori. Per tutti cerchiamo di chiarire la nostra posizione e le nostre intenzioni. I personal attualmente sul mercato sono caratterizzati da prestazioni e prezzi sensibilmente differenti, a seconda delle possibilità offerte e questo fatto, peraltro ovvio, determina già una prima discriminazione dal punto di vista hardware. A questo bisogna aggiungere il fatto che, essendo il BASIC un linguaggio non codificato da nessuno standard ufficiale (come invece ad esempio il FORTRAN o il COBOL), ogni costruttore ha utilizzato proprie istruzioni per rendere accessibili all'utente le funzioni meno usuali (grafica, colore, suono).

Infine bisogna tener conto che vengono utilizzati diversi microprocessori con architetture hardware completamente differenti e quindi ovviamente le locazioni di memoria dei vari sistemi non possono essere fisse.

Quindi, accertato che in molti casi, per accedere a determinate funzionalità è necessario scrivere programmi comprendenti istruzioni del tipo PEEK, POKE o PRINT CHR$ che si riferiscono a ben precise locazioni di memoria, si possono tuttavia

ECCO CHI TI AIUTERA' AD ANDARE D'AMORE E D'ACCORDO CON IL TUO NUOVO AMICO.



Il tuo concessionario IBM. Ti aiuterà a ottenere il massimo dal tuo Personal Computer IBM. Ti garantirà un'assistenza puntuale e un servizio all'altezza del nome IBM, che in tutto il mondo significa efficienza e affidabilità. Per una lunga e proficua amicizia fra te e il tuo Personal Computer IBM. Per acquisti superiori alle 20 unità puoi anche rivolgerti alle filiali IBM. E per ulteriori informazioni su eventuali punti di vendita che non compaiono sull'elenco, telefona a: 02/2172360 oppure 06/54864962.

ABRUZZI/MOLISE
Pescara - ITALDATA SRL - Via Tiburtina, 75 - Tel. 085.50843/54800
Campobasso - PUBLISISTEMI SRL - Via S. Antonio Abate, 236 - Tel. 0874.98144

BASILICATA
Potenza - I.P.E.S. SPA - Via Sanremo, 79 - Tel. 0971.43293

CALABRIA
Cosenza - CALIÒ SRL - Via N. Serra, 90 - Tel. 0984.32807

CAMPANIA
Cava dei Tirreni - METELLIANA SPA - Via Mandoli, 16 Tel. 089.463877
Napoli - POINTER SISTEMI SRL - Via A. De Gasperi, 45 Tel. 081.312312
Salerno - OMNIA SRL - C.so Garibaldi, 47 - Tel. 089.220366
S. Maria Capuavetere - GENERAL SYSTEMS SRL - Via Unità d'Italia, 21/23 - Tel. 0823.811100

EMILIA
Bologna
ABACO SAS - Via Bernini, 1 - Tel. 051.393274
CMB INFORMATICA SCRL - Via Arcoveggio, 74/10 - Tel. 051.323594
LUCKY SYSTEMS SRL - Via Farini, 33/A - Tel. 051.231569
SYSDATA ITALIA SPA - Via Massimo d'Azeglio, 58 - Tel. 051.330021
Carpi
DATA SRL - Via B. Peruzzi, 12 - Tel. 059.688090
UNIDATAX SRL - Via Biondo, 6 - Tel. 059.698355
Ferrara - MARKITALIA COMPUTERS SRL - Via Bologna, 84 - Tel. 0532.35867
Forlì - I.C.O.T. IMPIANTI SRL - Via Codazzi, 10 - Tel. 0543.723014
Imola - PALAZZO DONATO - Via Emilia, 23/A - Tel. 0542.29195
Piacenza - RCM COMPUTER SAS - C.so Vittorio Emanuele II, 96 - Tel. 0523.36263
Reggio Emilia
A.P.E.D. ELABORAZIONE DATI - Via Filippo Re, 7 - Tel. 0522.38721
MEMAR ELECTRONIC SRL - V.le Melato, 13 - Tel. 0522.94230
Rimini - HARD & SOFT SYSTEMS SRL - Via Valturio, 43 Tel. 0541.773343

LAZIO
Frosinone - SAIU ELETTRONICA SRL - Via Vado del Tufo, 85 - Tel. 0775.83093
Roma
CERVED SPA - Via Appia Nuova, 696 - Tel. 06.7940241
DATAOFFICE SPA - Via Sicilia, 205 - Tel. 06.4754568
ELEDRA 3S SPA - Via G. Valmarana, 63 - Tel. 06.8127324
GEDIN SRL - L.go D. De Dominicis, 7 - Tel. 06.432183
I.S.E.D. SPA - Via Tiburtina, 1236 - Tel. 06.4125851
JACOROSSI SPA - Via V. Brancati, 64 - Tel. 06.50091016
MEMORY COMPUTER SRL - Via Aureliana, 39 - Tel. 06.804592/4755736
NICA DIFF INF SRL - V.le Parioli, 40 - Tel. 06.872603
SAPES SRL - V.le Tito Livio, 12 - Tel. 06.3453536
VALDE ADEL SRL - P.zza S. Anastasia, 3 - Tel. 06.6786663
Viterbo - ITALBYTE SRL - V.le Trento - Pal. Garbini - Tel. 0761.221333

LIGURIA
Genova - DIFF.EL. SRL - Via XX Settembre, 31/4 - Tel. 010.586238

LOMBARDIA
Albino - NUOVA INFORMATICA SAS - Via Provinciale, 86 Comenduno - Tel. 035.751784
Assago - TRANSDATA SRL - Mi Fiori Pal. E3 Str. 1 - Tel. 02.8242460
Bergamo - SELTERING SPA - Via Verdi, 31 - Tel. 035.248256
Brescia
FIN-ECO SERVICE SRL - Via Pastrengo, 5 - Tel. 030.59055
MICROSELT SRL - Via Cipro, 33 - Tel. 030.224246
SELTERING SPA - Via Cipro, 33 - Tel. 030.220391
Como - BRUNO SRL - Via Rubini, 5 - Tel. 031.260538
Lecco - ZECCA UFFICIO SPA - Viale Dante, 14 - Tel. 0341.373291
Lodi - ZUCCHETTI SPA - C.so Mazzini, 39 - Tel. 0371.54827
Milano
AMUFFICIO SAS - Via Desenzano, 7 - Tel. 02.4080275
B.O.M. SAS - V.le Tunisia, 50 - Tel. 02.6598076
C.S.A. COMM. SRL - Via Farini, 82 - Tel. 02.6888433
DATA OPTIMATION SRL - Via Masaccio, 12 - Tel. 02.4987876
ECS ITALIA SRL - C.so Monforte, 15 - Tel. 02.780213
EDELEKTRON SRL - C.so Sempione, 39 - Tel. 02.3493603
ELEDRA 3S SPA - Viale Elvezia, 18 - Tel. 02.349751
GENERAL ELECTRIC INFORMATION SERVICES SPA V.le Regina Giovanna, 29 - Tel. 02.2870181
HOMIC PERSONAL COMPUTER SRL - Piazza De Angeli, 3 - Tel. 02.4988201
HUGNOT LUIGI LUCIANO - Via De Togni, 10 - Tel. 02.873190
IL NUOVO UFFICIO SISTEMI SNC - Via Priv. del Don, 2 - Tel. 02.8350870
MICROTECH SRL - Via F.lli Bronzetti, 20 - Tel. 02.733609
SIRIO SHOP SRL - Viale Certosa, 148 - Tel. 3010051
SOFTEC SRL - Viale Maino, 10 - Tel. 02. 7491196
STUDIO DI INFORMATICA S.D.I. SPA - Via G. Winckelmann, 1 - Tel. 02.4223305
Monza - EDICONSULT SRL - Via Rosmini, 3 - Tel. 039.389850
Pavia - I.T.C. INFORMATICA SRL - Strada Nuova, 86 - Tel. 0382.303201
S. Antonio Mantov. - ANTEK COMPUTER SAS - Via Manzoni, 49 - Tel. 0376.398759
Sondrio - G.P.D. SRL OFF. AUTOM. - V.le N. Sauro, 28 - Tel. 0342.218561
Varese
ELMEC SPA - Via Sebenico, 12 - Tel. 0332.264135
VEGA SPA - Via Silvestro Sanvito, 103 - Tel. 0332.229374
Vigevano - LOGICA INFORMATICA SRL - Via Montegrappa, 32 - Tel. 0381.81888
Vimercate - DATA PROGRES SRL - Via V. Emanuele, 44/A Tel. 039.667423
Vimodrone - OMEGA DATA SRL - Strada Padana Sup., 317 - Tel. 02. 2504121

MARCHE
Moie - S.E.D.A. SPA - P.zza S. Maria - Tel. 0731.70345/7603
Pesaro - COMPUTER & OFFICE SRL - Via Mazzini, 73 - Tel. 0721.64170

PIEMONTE
Alessandria - INFORMATICA SERVICE SRL - Via Isonzo, 63 - Tel. 0131.445817
Asti - HASTA DATI SNC - Via Silvio Morando, 6/A - Tel. 0141.216356
Biella
TEOREMA SRL - Via Losana, 9 - Tel. 015.24915
V.I.P. COMPUTERS SRL - Via Repubblica, 39 - Tel. 015.27106
Borgosesia - I.D.S. INF. DATA SYST. SRL - Viale Varallo, 157 - Tel. 0163.25327
Cuneo - SISTEMI SRL - Via Giolitti, 26 - Tel. 0171.55475
Genola - EUROSISTEMI SPA - Bivio S.S. 20/28 - Tel. 0172.68176
Torino
DIVERSIFICATE VENCO SRL - C.so Matteotti, 32A - Tel. 011.545525
PROGRAMMA SPA - C.so Svizzera, 185 - Tel. 011.746421
SISTEMI SPA - C.so Peschiera, 249 - Tel. 011.3358676
SOFTEC SRL - C.so San Maurizio, 79 - Tel. 011.8396444
Vercelli - ANALOG SNC - Via Dionisotti, 18 - Tel. 0161.61105

PUGLIE
Bari - PASED SRL - Via Calefati, 134/136 - Tel. 080.481488
Foggia - MASELLI PER L'UFFICIO - Via L. Zuppetta, 355A Tel. 0881.78014
Lecce - I.P.E.S. SPA - Via Oberdan, 29 - Tel. 0832.33904
Maglie - S.V.I.C. SRL - Via V. Emanuele, 121 - Tel. 0836.21604

SARDEGNA
Cagliari - C.D.S. SAS - Via Sonnino, 108 - Tel. 070.650756

SICILIA
Catania
ASIA COMPUTER SRL - Via S. Eupilio, 13 - Tel. 095.326944
COMPUTER SYSTEMS SRL - Via Ruggero di Lauria, 87 - Tel. 095.493777
Messina - SICIL FORNITURE SPA - Via Don Blasco, 75 Tel. 090.2923987
Palermo
SER.COM. ITALIA SRL - Via Sciuti, 180 - Tel. 091.261041
SI.PR.EL. SRL - Via Serradifalco, 145 - Tel. 091.577344
TESI SRL - Via E. Notarbartolo, 23 - Tel. 091.260549
Trapani - TESI SRL - Via Palmiero Abate, 2 - Tel. 0923.20026

TOSCANA
Empoli - SESA DISTRIBUZIONE SRL - Via XI Febbraio, 24/B - Tel. 0571.72148
Firenze
DATA COOP SCRL - Via di Novoli, 23/H - Tel. 055.416787
SESA DISTRIBUZIONE SRL - Lungarno Ferrucci, 19R - Tel. 055.6811652
Prato - C.C.S. SAS - Viale Repubblica, 298 - Tel. 0574.580222
Siena - SILOG SRL - Via Sicilia, 5 - Belverde - Tel. 0577.54085
Viareggio - DELPHI SRL - Via Aurelia Sud, 39 - Tel. 0584.31881

TRIVENETO
Bassano D/Grappa - C.P.E. - Piazzetta Poste, 9 - Tel. 0424.20395
Belluno - SCP. COMP. SYST. SRL - Via Feltre, 32 - Tel. 0437.70826
Bolzano - BOPAM SAS - Via C. Battisti, 32 - Tel. 0471.30113
Castelfranco Ven. - EDS SRL - Via S. Pio X, 154 - Tel. 0423.490178
Padova
CERVED ENGINEERING SPA - C.so Stati Uniti, 14 - Tel. 049.760733
S.I.C. ITALIA SRL - Via Fistomba, 8 - Tel. 049.45555
SYSTEM ROS SAS - P.zza De Gasperi, 14 - Tel. 049.38412
SO.GE.DA. SPA - Via Marsala, 29 - Tel. 049.655385/657386
S. Donà di Piave - COMPUTIME SRL - Piazza Rizzo, 63 Tel. 0421.2548
Trento
SEDA SAS - Via Sighere, 7/1 - Tel. 0461.984564
SiGe SNC - COMPUTER SHOP - Via Prato, 22 - Tel. 0461.25154
Treviso - INFORMATICA TRE SRL - Viale della Repubblica, 19 - Tel. 0422.65993
Trieste - DITTA MURRI - Via A. Diaz, 24/A - Tel. 040.733253
Udine
D.E.U. SRL - Via Di Prampero, 3/7 - Tel. 0432.204402
D.E.U. SRL - Via Tavagnacco, 89 - Tel. 0432.482086
Verona
PRAGMA SOFTWARE SRL - Via Carmelitani Scalzi, 20 - Tel. 045.24629
SEVER DI G. SERENI - Via Locatelli, 10 - Tel. 045.31331
Vicenza - ALFA DATA SRL - Via Milano, 110 - Tel. 0444.31865

UMBRIA
Perugia - PUCCIUFFICIO SNC - Via XX Settembre, 148/C Tel. 075.72992
Terni - DPS SRL - Via Pacinotti, 6 - Tel. 0744.58247

VALDAOSTA
Aosta
INFORMATIQUE SAS - Av. Du Cons. De Commis, 16 - Tel. 0165.2242

Per maggiori informazioni, compila e spedisci questo tagliando al tuo concessionario di zona.

Nome ________ Cognome ________

Via ________ N° ____

Cap ________ Città ________

IBM Italia
Distribuzione Prodotti srl

IBM

Servizio programmi

Per alcuni dei programmi pubblicati, *Personal Software* mette a disposizione dischi e nastri già registrati, realizzati in collaborazione con l'autore. Potete ottenerli in contrassegno, pagando direttamente al postino la cifra indicata, spedendo il tagliando pubblicato in fondo alla pagina.

N.	Sistema	Programmi	Supporto	pubblicato in *Personal Software* n.	Prezzo
1	Apple II+	La carta del cielo Collisione	floppy 5" DOS 3.3	3 pag. 83 3 pag. 93	30.000
2	TRS-80 mod. I	Backgammon	floppy 5" DOS 2.3	3 pag. 89	25.000
3	PET/CBM 3032/4032	Editor/Assembler in Basic	floppy 5" 3032/4032+3040/4040	2 pag. 33	40.000
4	Apple II+	Interi in precisione multipla Grafica 3D	floppy 5" DOS 3.3	4 pag. 17 4 pag. 47	40.000
5	PET/CBM 3032/4032	Gioco del calcio	floppy 5" 3032/4032+3040/4040	4 pag. 67	25.000
6	Apple II +	Pretty Printer Shape Table	floppy 5" DOS 3.3	5 pag. 27 5 pag. 58	30.000
7	Apple II +	Data base modulare	floppy 5" DOS 3.3	7 pag.	25.000

Spedire in busta chiusa a

PERSONAL SOFTWARE
Servizio Programmi
Via Rosellini 12
20124 Milano

Inviatemi i seguenti dischi di *Personal Software*

n. ____________________

per un totale di lire______________ + L. 2.000 come contributo fisso spese di spedizione che pagherò al postino alla consegna del pacco.

Cognome e nome

Indirizzo

Cap., Località

Firma

elencare alcuni accorgimenti per diminuire i disagi. Innanzitutto limitare al massimo l'utilizzo delle istruzioni anzidette anche a costo di dover inserire qualche istruzione in più. In secondo luogo, documentare accuratamente le locazioni di memoria a cui ci si riferisce, magari ampliando il discorso a quelle analoghe (ad esempio se mediante una PRINT CHR$ si seleziona un determinato colore, può essere interessante conoscere i codici degli altri colori accessibili, anche se in quel particolare programma non vengono utilizzati). Inoltre se di queste particolari istruzioni sono note le analoghe per altre macchine, è senz'altro utile segnalarlo, magari disegnando delle tabelle di conversione.

Per quanto riguarda la creazione di una rubrica contenente le locazioni di memoria dei vari sistemi, penso che sia difficilmente realizzabile, in quanto ogni costruttore adotta una politica differente: alcuni pubblicano elenchi dettagliati di tali locazioni nei manuali d'uso, mentre altri sono molto scarsi di informazioni.

Inoltre, funzioni che su alcune macchine sono disponibili mediante comandi BASIC immediati, su altre sono implementate in maniera differente o addirittura non ci sono.

Ricordo, comunque, che già da parecchi numeri abbiamo istituito la rubrica Conversioni, nata proprio per ridurre questi disagi, in cui sono bene accette ed attese le collaborazioni di tutti.

Meno giochi, più teoria

Spettabile Redazione,
scrivo per esporre alcune critiche che, vedo, accettate volentieri.

Nei primi numeri della rivista c'erano alcuni articoli teorici che in seguito andarono via via scomparendo.

È una grave lacuna.

Secondo me lasciate troppo spazio a listati (di giochi, generalmente) e poco ad articoli teorici.

Una rivista "in gamba" dovrebbe insegnare a creare buon software più che ad utilizzarlo.

Anche buona è la rubrica "piccoli annunci", alla quale vorrei appoggiarmi anch'io per scambiare software (progetto e listato) con altri produttori di software che la pensano come me.

Sperando vivamente che l'andamento del mercato non vi costringa altrimenti, colgo l'occasione per salutarvi.

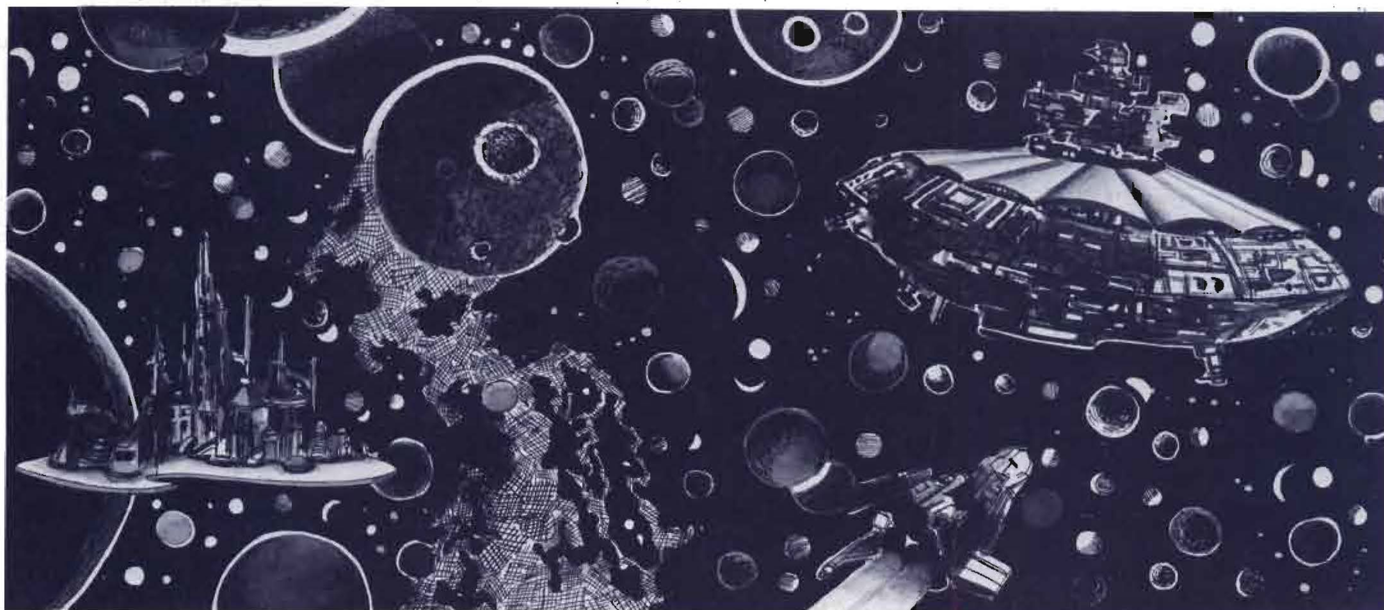
Paolo Bonafé
Favaro (VE)

Come probabilmente avrà notato, già in questo numero presentiamo alcuni articoli a carattere teorico e altri che costituiscono interessanti spunti per chiunque voglia avvicinarsi all'affascinante mondo dell'intelligenza artificiale. Anche nei prossimi numeri pubblicheremo articoli che sicuramente saranno molto apprezzati dai lettori che, come lei, preferiscono scriversi il proprio software.

Niente paura, comunque, per tutti quelli che invece premono per avere programmi già pronti per l'uso. Sappiamo che siete in molti, soprattutto tra i più giovani, e quindi continueremo a pubblicare programmi e giochi per i vari personal. La nostra intenzione è quella di accontentare un numero sempre crescente di appassionati, tenendo conto delle varie preferenze.

Per quanto riguarda la rubrica "piccoli annunci" abbiamo ricevuto moltissime inserzioni che cercheremo di pubblicare quanto prima, compatibilmente con lo spazio disponibile.

In questa rubrica rispondiamo alle lettere di carattere generale.
Scrivete a:
Personal Software
Via Rosellini, 12
20124 Milano



Conversione di programmi per ZX81 e ZX80 nuova ROM

— seconda parte —

Un esempio pratico di conversione

di *Bruno Del Medico*

In questa seconda puntata, viene presentata la conversione del Gioco del 15, *pubblicato sul n. 1 di* **Personal Software**. *Utilizzando le tecniche illustrate nella prima puntata, pubblicata sul numero scorso, l'autore documenta passo a passo l'evolvere del procedimento con delle osservazioni che vi saranno utili per le vostre conversioni. Nella terza ed ultima puntata verranno convertiti altri due programmi.*

La conversione di un programma può essere paragonata agevolmente ad una traduzione tra lingue diverse, per esempio italiano ed inglese. È facile tradurre termini come *casa* e *grande,* ma una frase del tipo: *a very large house* presenta già delle difficoltà perché una traduzione come: *una veramente grande casa* non è efficace.

Se poi pensate alle indicazioni di pesi e misure, o alle frasi fatte sul tipo di "lavarsene le mani" cominciate a capire la complessità dell'argomento e vi rendete conto che la traduzione letterale non è quasi mai sufficiente ma deve essere accompagnata da una interpretazione. Molto spesso diventa necessario ricorrere a perifrasi o giri di parole per ottenere un risultato soddisfacente.

Nella conversione di programmi per lo ZX81 questo accade ancora più spesso perché il suo BASIC è generalmente limitato rispetto a quello di altri computer, e presenta grosse diversità specie per quanto riguarda la grafica.

Ecco perché la conversione di un programma che non sia molto semplice va sempre eseguita per passi successivi:

a) il primo passo consiste nello studio del programma, per individuare le principali routine, subroutine e modalità di funzionamento, si può così capire quali sono i passi tecnicamente intraducibili,e decidere quale tipo di forma programmatoria alternativa si deve usare per ottenere un risultato analogo;

b) bisogna poi effettuare la conversione meccanica del programma, rendendolo sintatticamente compatibile allo ZX81, ma scorretto nel significato;

c) infine si carica il programma e lo si fa girare, eliminando in una prima fase tutti gli errori più evidenti, come per esempio quelli di tipo 5 (non c'è più spazio sullo schermo) o di tipo 3 (indice fuori dal range stabilito); si fa poi in modo che la routine grafica compia il suo dovere stampando le cose giuste al posto giusto, magari procedendo per tentativi. A questo punto il programma girerà in modo soddisfacente, e si potrà migliorarne l'impostazione generale.

Conversione di un programma per Commodore: il Gioco del 15

Questo programma è apparso sul numero 1 di **Personal Software**, che consiglio di avere sotto mano per seguire sul listato originale le tecniche usate nella conversione.

Si deve innanzitutto ragionare sul tipo di programma che ci si accinge a convertire: siccome *Il gioco del quindici* si svolge su una scacchiera 4*4, dovranno sicuramente essere presenti:

a) un vettore, o matrice, che contenga i 16 elementi del gioco: i 15 numeri più lo spazio vuoto che viene utilizzato per gli spostamenti. Qualsiasi programma che si svolga su una scacchiera, salvo rare eccezioni, usa un vettore o una matrice per contenere i dati relativi agli elementi mobili, pedine o altro;

b) una routine grafica che stampa la scacchiera per velocizzare l'esecuzione è divisa in una parte che stampa la scacchiera vera e propria (parte fissa), ed una parte che disegna nelle caselle gli elementi del vettore corrispondente. Dopo ogni mossa si ottiene l'aggiornamento della situazione usando solo la seconda parte. Nei programmi più elementari la scacchiera non viene disegnata e si stampano gli elementi del vettore in un certo ordine. Per esempio:

8	4	2
1	5	6
7	3	9

Le routine grafiche si riconoscono facilmente perché sono costituite da un ciclo FOR NEXT o, assai più spesso, da più cicli concatenati.

c) – Un ciclo che inizializza gli elementi del vettore, o due cicli concatenati se gli elementi del gioco sono contenuti in una matrice,
– una routine (con relativa istruzione INPUT) che accetta la mossa del giocatore, la controlla e modifica di conseguenza i contenuti della matrice o del vettore,
– una routine che determina la mossa dell'avversario. Quando l'avversario è il computer la complessità di questa routine è generalmente indicativa della maggiore intelligenza di gioco. Nel caso in esame il giocatore non ha avversari.

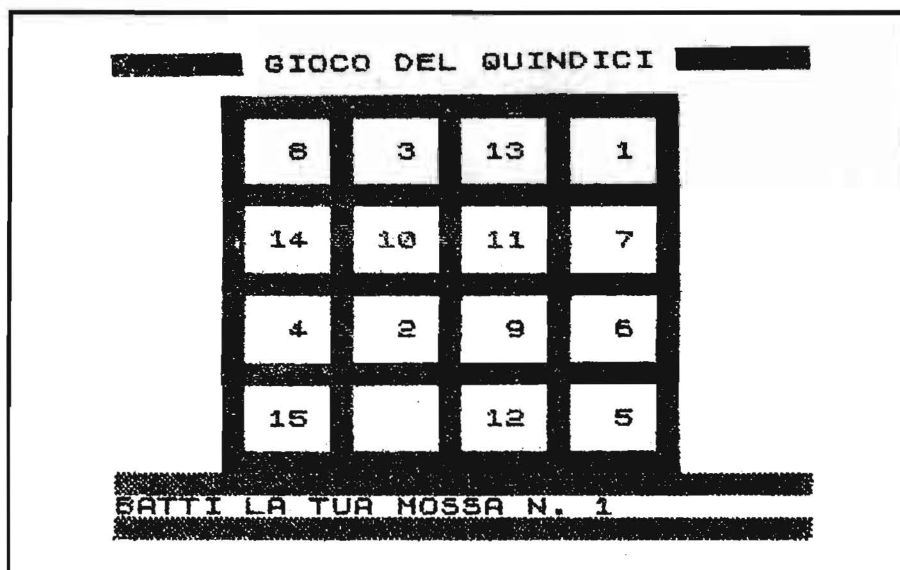


Figura 1. *Quadro iniziale del* Gioco del 15 *in caso di errore del giocatore sono possibili 2 messaggi: "Solo i numeri da 1 a 15 prego" e "Mossa non permessa". Si noti che il primo viene stampato sulla linea 23 dello schermo, che normalmente non può essere utilizzata ma diventa accessibile con l'istruzione POKE 16418, 0.*

Esaurite queste considerazioni basta leggere attentamente il listato del *Gioco del quindici*, cercando tutti i GOSUB e localizzando le subroutine:

a) la linea 160 contiene un GOSUB 650. Quindi alla linea 650 inizia una subroutine che evidentemente termina al primo RETURN successivo. È importante identificare per ogni subroutine che funzione svolge;

b) con lo stesso metodo del punto a) si localizzano altre subroutine:
– 580/640, richiamata dalla linea 170;
– 800/830, richiamata dalle linee 180, 200, 220, 270 e moltissime altre;
– 530/570, richiamata dalle linee 340, 360, 380 e 400.

Dunque il programma in oggetto è ben ordinato e si divide in due blocchi:

a) il blocco delle subroutine, tutte raccolte nelle linee tra la 530 e la 830;

b) il corpo principale del programma, che va dalla linea 100 alla linea 520.

La linea 520 è un GOTO 170, infatti solo la parte che va dalla linea 170 alla linea 520 deve essere ripetuta ad ogni mossa.

Le linee dalla 100 alla 160 delimitano la parte introduttiva, nella quale vengono inizializzati gli elementi del gioco (per esempio, dove si mischiano le carte o si dispongono le pedine sulla scacchiera). In questa parte introduttiva viene sicuramente inizializzato il vettore o matrice che dovrà contenere i 15 numeri del gioco. Si noti inoltre che il blocco di programma che va dalla linea 170 alla 520 richiama tutte le subroutine individuate in precedenza, eccetto la 650. Questa viene richiamata una sola volta, nella parte introduttiva, dalla linea 160.

La parte introduttiva e le subroutine

La linea 100 inizializza una matrice numerica di nome N a 2 dimensioni (4, 4) e composta da 16 elementi che possono essere razionalmente rappresentati come in tabella 1.

N (1, 1) = 0	N (2, 1) = 0	N (3, 1) = 0	N (4, 1) = 0
N (1, 2) = 0	N (2, 2) = 0	N (3, 2) = 0	N (4, 2) = 0
N (1, 3) = 0	N (2, 3) = 0	N (3, 3) = 0	N (4, 3) = 0
N (1, 4) = 0	N (2, 4) = 0	N (3, 4) = 0	N (4, 4) = 0

Tabella 1. *Matrice a 2 dimensioni composta da 16 elementi.*

Si deve ricordare che i sedici elementi della matrice N non possono rimanere uguali a zero, ma devono avere valori da 0 a 15.

Ciò viene ottenuto con la routine 110/140, illustrata nel riquadro "Inizializzazione di un vettore".

Si incontra poi una istruzione PRINT CHR$ (144), spiegata nella tabella 2 insieme ad altre istruzioni speciali del VIC 20, che sono generalmente adatte per tutti i Commodore (PRINT CHR$ (144)). Siccome nello ZX81 abbiamo solo la stampa in nero, questa istruzione può essere ignorata.

Ora che la parte iniziale del programma è stata compresa nel suo complesso, la si può convertire secondo le indicazioni riportate nel numero precedente. Si ricaverà qualcosa di simile alla parte di programma compresa tra le linee 99 e 160 del listato 1.

Si aggiunge la linea:

```
99 LET MOSSE = 1
```

perché la variabile MOSSE, incrementata di 1 nella linea 480, non è stata inizializzata in precedenza.

La subroutine 650 è l'unica richiamata nella parte introduttiva, e viene eseguita una volta sola.

La sua prima istruzione è PRINT CHR$ (147), equivalente a CLS, che pulisce lo schermo.

Si incontrano poi ben due istruzioni PRINT CHR$ (18), che predispongono la scrittura in modo inverso (la seconda non disattiva la prima). Tutti i PRINT presenti in questa subroutine e nel programma, siccome non si incontrano istruzioni PRINT CHR$ (146) che disattivano l'inverse, stampano spazi inversi.

Nella subroutine si possono distinguere due loop: il primo è concatenato e va dalla linea 670 alla 710; l'altro è un ciclo semplice e va dalla linea 730 alla 770.

Nel primo loop il ciclo più esterno FOR RIGA viene eseguito 5 volte (4 TO 20 STEP 4) ed ogni volta il ciclo più interno FOR H disegna 21 spazi inversi in senso orizzontale, partendo dalla colonna 10 dello schermo (PRINTTAB 10). In pratica questo ciclo disegna 5 linee orizzontali che vanno dalla colonna 10

Listato 1. Gioco del quindici *prima passata. Questo listato rappresenta una fase intermedia della conversione del programma* Gioco del quindici. *Questa versione gira solo se vengono apportate le modifiche descritte nel testo. Potete trovare la versione definitiva nel listato 2.*

```
  1 REM GIOCO 15 PRIMA PASSATA
 99 LET MOSSE=0
100 DIM N(4,4)
110 FOR K=1 TO 15
120 LET I=INT (RND*4+1)
121 LET J=INT (RND*4+1)
130 IF N(I,J)<>0 THEN GOTO 120
140 LET N(I,J)=K
141 NEXT K
160 GOSUB 550
170 GOSUB 580
180 LET X=22
181 GOSUB 800
190 PRINT "▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒"
200 LET X=23
201 GOSUB 800
210 PRINT "▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒"
220 LET X=23
221 GOSUB 800
230 PRINT "BATTI LA TUA MOSSA"
231 INPUT M$
240 IF LEN M$<1 THEN GOTO 270
250 IF M$="F" THEN STOP
260 LET M=VAL "M$"
261 IF M>0 AND M<16 THEN GOTO 3
00
270 LET X=22
271 GOSUB 800
280 PRINT "ERRORE.USA NUMERI DA
1 A 15"
290 GOTO 200
300 LET X=23
301 GOSUB 800
310 FOR I=1 TO 4
311 FOR J=1 TO 4
320 IF N(I,J)=M THEN GOTO 340
330 NEXT J
331 NEXT I
340 LET II=I+1
341 LET JJ=J
342 GOSUB 530
350 IF FLAG=1 THEN GOTO 460
360 LET II=I-1
361 LET JJ=J
362 GOSUB 530
370 IF FLAG=1 THEN GOTO 460
380 LET II=I
381 LET JJ=J+1
382 GOSUB 530
390 IF FLAG=1 THEN GOTO 460
400 LET II=I
401 LET JJ=J-1
402 GOSUB 530
410 IF FLAG=1 THEN GOTO 460
420 LET X=22
421 GOSUB 800
430 PRINT "▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒"
440 LET X=22
441 GOSUB 800
442 PRINT "MOSSA NON PERMESSA"
```

Seguito listato 1.

```
450 GOTO 200
460 LET N(I,J)=0
461 LET N(II,JJ)=M
470 LET X=22
471 GOSUB 800
480 LET MOSSE=MOSSE+1
490 LET X=4
491 GOSUB 800
492 PRINT TAB 25;"MOSSE";MOSSE
510 LET X=2+J*4
511 GOSUB 800
512 PRINT TAB (7+I*5);"█";
520 GOTO 170
530 LET FLAG=0
540 IF II<1 OR II>4 OR JJ<1 OR
JJ>4 THEN GOTO 570
550 IF N(II,JJ)<>0 THEN GOTO 57
0
560 LET FLAG=1
570 RETURN
580 FOR I=1 TO 4
581 FOR J=1 TO 4
590 IF N(I,J)=0 THEN GOTO 630
600 LET X=2+J*4
601 GOSUB 800
610 PRINT TAB (6+I*5-LEN (STR$
(N(I,J)))+1);
620 PRINT N(I,J)
630 NEXT J
631 PRINT
632 NEXT I
640 RETURN
650 CLS
660 PRINT TAB 7;"GIOCO DEL 15"
670 FOR R=4 TO 20 STEP 4
690 LET X=R
691 GOSUB 800
692 PRINT TAB 10;
700 FOR H=10 TO 30
701 PRINT "█";
702 NEXT H
710 PRINT
711 PRINT
712 NEXT R
713 PRINT
730 FOR L=5 TO 19
731 LET X=L
732 GOSUB 800
740 PRINT TAB 10;"█";TAB 15;"█"
;TAB 20;"█";TAB 25;"█";TAB 30;"█
"
770 NEXT L
771 PRINT
772 PRINT TAB 1;
780 LET X=23
781 GOSUB 800
790 PRINT "BATTI F PER FINIRE"
791 RETURN
800 PRINT AT 0,0;
810 FOR Y=1 TO X-1
820 PRINT AT Y,0;
830 NEXT Y
831 RETURN
```

alla colonna 30: che risultano spostate rispetto al centro dello schermo dello ZX81 che ha 32 colonne, ma non per il Commodore che ne ha 40.

Comunque il disegno, anche se fuori centro, entra nello schermo dello ZX81 e per il momento lo si può lasciare inalterato, si cambia invece il nome della variabile di controllo RIGA, che deve essere composto da un solo carattere.

Le cinque righe nere disegnate dal ciclo concatenato sono spaziate di 3 posizioni perché vengono tracciate lungo le linee di schermo 3, 7, 11, 15 e 19.

Il metodo usato per ottenere la spaziatura di 3 linee è abbastanza ingegnoso ed utilizza la subroutine 800 già impiegata per selezionare la linea di stampa, cioè per spostare il cursore da una posizione all'altra. All'inizio della subroutine si ha l'istruzione PRINT CHR$ (19) che equivale a PRINT AT 0, 0. Segue il ciclo FOR Y che viene eseguito tante volte quanto è il valore di X–1, ad ogni ripetizione viene eseguita l'istruzione PRINT CHR$ (17) che abbassa di un posto la posizione di stampa (cioè abbassa il cursore di una riga). In pratica ogni volta che il ciclo FOR Y viene ripetuto, la posizione di stampa diventa: 1, 0; 2, 0; 3, 0;... e così via, tante volte quanto è il valore di X–1.

Siccome in molte linee, X assume valori fino a 23, si avranno dei problemi con la dimensione dello schermo relativamente alle linee in fondo, dal momento che lo ZX81 ha 22 linee utilizzabili mentre il Commodore ne ha 24.

Nella subroutine 650 la variabile X è posta uguale a RIGA e varia da 4 a 20, le cinque linee orizzontali vengono dunque disegnate nelle linee dalla 3 alla 19.

Il secondo loop, FOR L, va da 5 a 19 e viene eseguito 15 volte, ogni volta disegna nella linea X, alle colonne 10, 15, 20, 25 e 30, un quadratino nero. La posizione di stampa si abbassa di una linea ad ogni ripetizione del ciclo perché X varia, in questo modo la subroutine 650 nel suo complesso disegna 5 linee orizzontali e 5 verticali, le quali compongono un reticolo facilmente identificabile con la scacchiera che dovrà contenere i 15 numeri.

Il Commodore ricorre alla subroutine 800 per spostare la posizione del cursore in una linea diversa, mentre lo ZX81, modesto ma efficace, utilizza l'istruzione PRINT AT.

Una volta individuata la subroutine che stampa la scacchiera, si riconosce che quella che stampa gli elementi della matrice numerica nei corrispondenti spazi è l'altra subroutine contenente un ciclo concatenato, la 580/640.

Fortunatamente la scacchiera entra nello schermo dello ZX81, perché occupa le posizioni comprese nelle seguenti coordinate:

3, 10	3, 30
19, 10	19, 30

Quando il programma girerà bene in tutti i suoi aspetti, si potrà centrare meglio la scacchiera, ponendola per esempio alle coordinate:

2, 6	2, 26
18, 6	18, 26

In questo modo in fondo allo schermo ci saranno almeno tre linee per eventuali messaggi mentre nella disposizione attuale rimangono in fondo allo schermo due sole linee e sicuramente questo causerà errori di tipo 5 se si farà girare la prima passata del programma; tuttavia saranno inconvenienti facilmente rimediabili.

Il corpo principale del programma

Il corpo del programma contiene l'algoritmo del gioco, sarebbe interessante discuterne le modalità di svolgimento; ma non interessa ai fini di questo articolo. Del resto gli algoritmi di calcolo sono i più facili da convertire e sono sufficienti le tecniche esposte nell'articolo precedente.

Bisogna notare che il programma è sempre sotto l'effetto della istruzione PRINT CHR$ (18) e le linee di programma sul tipo della 190 disegnano spazi inversi e non spazi bianchi. Ovviamente i cicli che stampano linee orizzontali come queste devono andare da 1 a 32 e non da 1 a 40.

Dopo aver capito la logica di base di questo programma si può effettuare una prima conversione: l'obiettivo è di rendere il programma caricabile sullo ZX81.

In questo modo lo si fa girare, gli inevitabili errori potranno essere corretti sperimentalmente e non attraverso congetture effettuate sul listato.

La prova pratica

Terminata la prima conversione del programma, che appare nel listato 1, lo si carica sullo ZX81 e lo si registra due volte, onde evitare brutte sorprese, e si dà il RUN.

Listato 2. Gioco del quindici. *Questo programma gira su ZX81 e ZX80/8K. Occorre riordinare in una scacchiera 4×4 i numeri da 1 a 15 presentati alla rinfusa, utilizzando per gli spostamenti l'unica casella libera. Il gioco è piacevole perché sufficientemente veloce. L'output di questo programma è quello della figura 1.*

```
  3 REM GIOCO DEL QUINDICI
  4 REM
  5 REM
 10 GOSUB 9000
100 DIM N(4,4)
101 LET MOSSE=1
110 FOR U=1 TO  15
120 LET I=INT (RND*4)+1
122 LET J=INT (RND*4)+1
130 IF N(I,J)<>0 THEN GOTO 120
140 LET N(I,J)=U
142 NEXT U
160 GOSUB 650
170 GOSUB 580
190 PRINT AT 19,0;""
195 PRINT ""
200 PRINT AT 20,0;"BATTI LA TUA
MOSSA N. ";MOSSE
210 PRINT ""
232 INPUT M$
240 IF CODE M$<29 OR CODE M$>37
THEN GOTO 270
241 IF LEN M$=1 THEN GOTO 250
242 IF CODE M$(2)<28 OR CODE M$
(2)>33 THEN GOTO 270
260 LET M=VAL M$
262 IF M>0 AND M<16 THEN GOTO 3
00
270 POKE 16418,0
272 PRINT AT 23,0;"SOLO NUMERI
DA 1 A 15 PREGO"
275 PAUSE 400
276 POKE 16436,255
280 POKE 16418,2
290 GOTO 232
300 FOR I=1 TO 4
312 FOR J=1 TO 4
320 IF N(I,J)=M THEN GOTO 340
330 NEXT J
332 NEXT I
340 LET II=I+1
342 LET JJ=J
344 GOSUB 530
350 IF FLAG=1 THEN GOTO 460
360 LET II=I-1
362 LET JJ=J
364 GOSUB 530
370 IF FLAG=1 THEN GOTO 460
380 LET II=I
382 LET JJ=J+1
384 GOSUB 530
390 IF FLAG=1 THEN GOTO 460
400 LET II=I
402 LET JJ=J-1
404 GOSUB 530
410 IF FLAG=1 THEN GOTO 460
430 PRINT AT 19,0;""
444 PRINT "MOSSA NON PERMESSA R
IPROVA"
450 GOTO 232
460 LET N(I,J)=0
462 LET N(II,JJ)=M
480 LET MOSSE=MOSSE+1
490 IF N(3,4)=15 AND N(2,4)=14
AND N(1,4)=13 THEN GOTO 492
491 GOTO 510
```

Seguito listato 2.

```
 492 IF N(4,3)=12 AND N(3,3)=11
AND N(2,3)=10 AND N(1,3)=9 THEN
GOTO 494
 493 GOTO 510
 494 IF N(4,2)=8 AND N(3,2)=7 AN
D N(2,2)=6 AND N(1,2)=5 THEN GOT
O 496
 495 GOTO 510
 496 IF N(4,1)=4 AND N(3,1)=3 AN
D N(2,1)=2 AND N(1,1)=1 THEN GOT
O 5000
 510 LET X=J*4
 514 PRINT AT X,(2+I*5);"  ";
 520 LET X=JJ*4
 524 PRINT AT X,(4+II*5-LEN M$);
M
 528 GOTO 160
 530 LET FLAG=0
 540 IF II<1 OR II>4 OR JJ<1 OR
JJ>4 THEN GOTO 570
 550 IF N(II,JJ)<>0 THEN GOTO 57
0
 560 LET FLAG=1
 570 RETURN
 580 FOR I=1 TO 4
 582 FOR J=1 TO 4
 590 IF N(I,J)=0 THEN GOTO 630
 600 LET X=J*4
 610 PRINT AT X,(3+I*5-LEN (STR$
(N(I,J)))+1);
 620 PRINT N(I,J)
 630 NEXT J
 632 PRINT
 634 NEXT I
 640 RETURN
 650 CLS
 660 PRINT "██████ GIOCO DEL QUI
NDICI ██████"
 670 FOR S=2 TO 18 STEP 4
 690 LET X=S
 694 PRINT AT X,6;
 700 FOR H=6 TO 25
 702 PRINT "█";
 704 NEXT H
 710 PRINT
 712 PRINT
 714 NEXT S
 716 PRINT
 730 FOR L=2 TO 18
 732 LET X=L
 740 PRINT AT X,5;"█";TAB 10;"█"
;TAB 15;"█";TAB 20;"█";TAB 25;"█
"
 770 NEXT L
 772 PRINT
 792 RETURN
5000 GOSUB 9000
5010 PRINT AT 10,0;"COMPLIMENTI"
5012 PRINT
5014 PRINT "HAI VINTO IN ";MOSSE
;" MOSSE"
5020 INPUT U$
5030 CLS
5040 RUN
9000 CLS
9010 PRINT "████████████████████
████████████"
9020 PRINT TAB 7;"GIOCO DEL QUIN
DICI"
9030 PRINT "████████████████████
████████████"
9040 PRINT
9050 PRINT
9060 RETURN
```

Dopo pochi secondi, sullo schermo appare la scacchiera come la si era immaginata esaminando la subroutine 650. Nelle caselle non ci sono però i numeri, ed in fondo allo schermo appare il previsto ma non temuto messaggio di errore:

5/820

Evidentemente qualcuna delle X inizializzate al valore di 23 ha indotto la subroutine 800 a posizionare il cursore nella linea 22.

Per evitare questi problemi bisogna cancellare tutte le linee che ricorrono all'uso di X > = 22 per determinare la posizione di stampa. Successivamente si deve verificare quali messaggi vengono stampati fuori posto, e correggerne la posizione inserendo delle istruzioni PRINT AT.

Tutte le linee dalla 200 alla 221, dalla 420 alla 441, le 470 e 471, 780 e 781, 790, 190, 180 e 181 devono essere cancellate. Queste tentavano di scrivere qualcosa sotto la scacchiera, dove sono utilizzabili solo 2 linee; una occupata da un PRINT eseguito nella subroutine 650, l'altra dalla domanda BATTI LA TUA MOSSA. Adesso il programma gira anche se con alcuni difetti:

a) il messaggio contenente il numero di mosse fatte viene stampato sulla linea 3 e si sovrappone all'angolo destro in alto della scacchiera. Si rimedia modificando così la linea 492:
   492 PRINT AT 2, 20;
   "MOSSA"; MOSSE

b) quando si fa una mossa, il numero battuto sulla tastiera va normalmente ad occupare la casella vuota. Però la posizione occupata in precedenza, che dovrebbe venire coperta da due spazi bianchi, viene invece occupata da due spazi neri. Si rimedia modificando così la linea 512:
   512 PRINT TAB (7+1*5);
   "due spazi bianchi"

c) se si introduce un numero errato o una mossa non consentita, il computer tenta di stampare il messaggio di errore sotto la scritta BATTI LA TUA MOSSA, cioè nella linea 22, basta modifi-

Inizializzazione di un vettore

Con l'istruzione:

DIM P(3)

inizializziamo un VETTORE NUMERICO, cioè un gruppo di variabili tutte con nome P. Per distinguerle tra loro le chiamiamo: P(1), P(2) e P(3). Appena inizializzati i tre elementi del vettore P valgono O. Per assegnare un valore diverso, per esempio da 1 a 3 possiamo scrivere:

```
LET P(1) = 1
LET P(2) = 2
LET P(3) = 3
```

oppure

```
10 FOR K = 1 TO 3
20 LET P(K) = K
30 NEXT K
```

Ammettiamo di voler assegnare ad un vettore P, composto da 90 elementi, i valori interi da 1 a 90, possiamo farlo con un ciclo FOR K come il precedente. Immaginiamo ora di volergli attribuire valori da 1 a 90, ma ALLA RINFUSA, per esempio facendo in modo che l'elemento P(90) valga 1, e l'elemento P(1) valga 43:

```
10 DIM P(90)
20 FOR K = 1 TO 90
30 LET P(K) = INT (RND * 90) + 1
40 NEXT K
```

In questo modo però si potranno avere molti elementi con valori uguali tra loro, perché la linea 40 potrebbe generare per esempio diverse volte il numero 80 e neppure una volta il numero 79.

Se si vuole avere un vettore P contenenti TUTTI i numeri da 1 a 90 sistemati alla rinfusa, si deve usare una routine come questa:

```
10 DIM P(90)
15 DIM Q(90)
20 FOR K = 1 TO 90
30 LET NUM = INT (RND * 90) + 1
32 IF Q(NUM) = 1 THEN GOTO 30
34 LET Q(NUM) = 1
36 LET P(K) = NUM
40 NEXT K
```

Vediamo come agisce questa routine, tenendo presente che all'inizio tutti gli elementi di P e tutti quelli di Q valgono zero:

```
K=1  Linea 30: NUM = 23
     linea 32: Q(32) = 1? NO continua
     linea 34: Q(23) = 1
     linea 36: P(1) = 23
K=2  Linea 30: NUM = 25
     linea 32: Q(25) = 1? NO continua
     linea 34: Q(25) = 1
     linea 36: P(2) = 25
K=3  Linea 30: NUM = 23
     linea 32: Q(23) = 1? SI. RITORNA ALLA LINEA 30.
```

Il programma torna alla linea 30 ogni volta che viene selezionato un numero già scelto in precedenza.

La seguente routine può essere usata nel gioco della TOMBOLA o del LOTTO ed estrae i 90 numeri evitando di estrarre due volte lo stesso numero:

```
10 DIM P(90)
15 FOR K = 1 TO 90
20 LET P(K) = K
30 NEXT K
40 FOR K = 1 TO 90
50 LET NUM = INT (RND * 90) + 1
60 IF P(NUM) = 0 THEN GOTO 50
70 LET P(NUM) = 0
80 LET NUM = NUMERO ESTRATTO
90 NEXT K
```

A tutti quelli che vogliono approfondire le tecniche di programmazione in BASIC consiglio il mio volume Sviluppiamo software con il nostro ZX81, Edizioni Clup, Milano.

care le linee di programma contenenti messaggi di errore introducendo una istruzione PRINT AT 20, 0. Per esempio:

```
442 PRINT AT 20, 0;
"MOSSA NON PERMESSA"
```

Adesso i messaggi vengono stampati nell'unica linea disponibile, quella appena sotto la scacchiera e appena sopra la richiesta di mossa. In questo modo però il messaggio non viene più cancellato, si deve introdurre subito dopo l'input una linea:

```
232 PRINT "32 spazi"
```

Questa linea viene eseguita appena il giocatore ripete la mossa e cancella il messaggio di errore, il fatto che venga eseguita OGNI VOLTA che il giocatore fa una mossa, non ha importanza.

A questo punto si può essere già soddisfatti del lavoro eseguito: in due passate si è ottenuto un programma che gira, e per i meno esigenti può andare bene anche così.

I più esigenti potranno effettuare anche una terza passata per correggere i difetti che ancora sussistono:

a) tutto lo schermo è disegnato in modo asimmetrico rispetto al suo asse centrale;

b) la verifica dell'input, che era sufficientemente efficace per il Commodore, non lo è più per il Sinclair. Infatti nello ZX81, se il giocatore batte la mossa premendo una lettera anziché un numero, il computer si blocca;

c) il gioco è lento sullo ZX81, perché ogni volta che viene eseguita una mossa il computer ristampa tutta la matrice. Inoltre il computer non riconosce la fine del gioco. È previsto che, quando il giocatore termina di sistemare i 15 numeri, prema F per finire. Si è cercato di porre rimedio a tutti questi inconvenienti nel programma del listato 2. La videata principale è visibile nella figura 1 e si noti che la scacchiera è stata centrata.

Se il giocatore batte una mossa non concessa (es. 2) il computer

Tabella 2. *Istruzioni particolari usate su Commodore.*

CHR$ (5)	Predispone il testo in bianco.
CHR$ (28)	Predispone il testo in rosso.
CHR$ (30)	Predispone il testo in verde.
CHR$ (31)	Predispone il testo in blu.
CHR$ (144)	Predispone il testo in nero.
CHR$ (156)	Predispone il testo in viola.
CHR$ (158)	Predispone il testo in giallo.
CHR$ (159)	Predispone il testo in azzurro.
CHR$ (18)	Attiva l'INVERSE.
CHR$ (146)	Disattiva l'INVERSE.
CHR$ (13)	Sostituisce il RETURN.
CHR$ (17)	Fa abbassare il cursore di una riga.
CHR$ (19)	Sposta il cursore nell'angolo in alto a sinistra.
CHR$ (29)	Sposta il cursore a destra di una colonna.
CHR$ (32)	Produce uno spazio.
CHR$ (141)	Equivale a SHIFT e RETURN.
CHR$ (145)	Fa alzare il cursore di una riga.
CHR$ (147)	Pulisce lo schermo e manda il cursore in alto a sinistra.
CHR$ (157)	Sposta il cursore a sinistra di una posizione.
CHR$ (160)	Produce uno spazio + SHIFT.
SPC(x)	Fa spaziare di x posizioni senza cancellare.
TAB(x)	Fa andare alla colonna x. Se x supera 21 si passa alla linea successiva.
CHR$ (14)	Fa passare al set maiuscolo/minuscolo.
CHR$ (142)	Fa passare al set grafico.
ASC(X$)	Equivale a CODE X$.
FRE(x)	Fornisce la quantità di byte liberi in memoria.
POS(x)	Restituisce un numero da 0 a 21, che rappresenta la colonna su cui si trova il cursore.

risponde con un messaggio di errore, ma siccome in fondo allo schermo non c'è più spazio il messaggio viene stampato sulla linea 23 (che normalmente non è accessibile) dalla routine 270/290. La scritta scompare dopo 8 secondi (o anche prima se il giocatore preme un tasto qualsiasi), ed il programma torna ad eseguire l'input alla linea 232.

Le linee 490/496 riconoscono la situazione di gioco terminato. Questa versione è sufficientemente veloce per essere anche piacevole.

Per ottenere una maggiore velocità, la scacchiera ed il vettore con i quindici numeri vengono disegnati una sola volta. Successivamente il computer aggiorna solo la posizione relativa al numero spostato. Effettuata una mossa, dopo circa un secondo si ottiene il READY per la mossa successiva. ■

Introduzione alla intelligenza artificiale

— prima parte —

Un contributo alla "istruzione" del vostro personal

di *Bruno Del Medico*

Tutti gli utilizzatori di personal possono sicuramente dividere la propria biblioteca di giochi in due sezioni:

- programmi nei quali il computer svolge un ruolo di arbitro, o crea delle situazioni funzionali allo svolgimento del gioco, ma casuali;
- programmi nei quali il computer svolge in modo completo il ruolo di avversario, gioca contro l'operatore e lo fa con una certa intelligenza, oppure coordina il comportamento intelligente dell'avversario di turno.

Scopo di questo articolo è di entrare nei dettagli dei programmi della seconda categoria: capire come fa il computer ad essere tanto abile da risultare talvolta imbattibile.

Parleremo quindi di IA (Intelligenza Artificiale). Presenterò delle routine intelligenti, spiegandone nel dettaglio le modalità di funzionamento. Successivamente presenterò alcuni programmi basati sulla tecnica del processo di apprendimento. (In questi programmi abbiamo uno svolgimento simile a quello illustrato nei diagrammi della figura 1).

Tutti i programmi presentati in questo articolo girano su ZX81, e, con le modifiche suggerite alle figure 2 e 3, anche su Spectrum e su ZX80 con ROM da 8 Kbyte.

Tobia, bruco poco astuto

Faremo i primi passi nel campo della IA immaginando di trovarci proprio in un campo, e di osservare le peripezie di un simpatico bruco, cui affibbiamo il nome di Tobia Smith. Forse non possiede neppure un milligrammo di cervello, e sicuramente lo impiega tutto in una sola attività: la ricerca del cibo. Non sempre lo fa con successo. Se caricate il listato 1, lo vedrete affannarsi avanti e indietro, (la stringa A$), su e giù nel suo territorio, mentre il cibo (l'asterisco inverso) se ne sta in un angolo senza manifestare eccessive preoccupazioni. Alcune volte Tobia lo trova, altre volte no ed allora purtroppo... muore di fame.

Questo accade quando il ciclo FOR K (linea 220) viene eseguito tutte le 500 volte. L'istruzione 240:

```
PEEK 16398 + 256 *
PEEK 16399
```

fornisce l'indirizzo del display file (memoria di schermo) relativo al punto che sta per essere stampato, cioè L, C (linea 230).

Guardando la linea 300 potete vedere che il punto L, C viene occupato dal bruco. Il computer si accorge che il bruco sta per soprastampare il cibo (quindi lo sta catturando) se il contenuto dell'indirizzo letto con la precedente istruzione PEEK è 151,

numero di codice dell'asterisco inverso. Poiché i contenuti degli indirizzi si leggono con l'istruzione PEEK, la linea 240:

PEEK (PEEK 16398 + 256 * PEEK 16399)

indica:

IL CONTENUTO (dell'indirizzo che puoi ottenere leggendo il contenuto dei byte 16398 e 16399).

Notate bene la sequenza delle linee nel listato 1:

- la linea 230 stabilisce che qualcosa deve essere scritto nella posizione L,C; da quel momento L,C rappresenta la posizione di stampa.
- La linea 240 legge nel display file il contenuto della posizione di stampa, cioè stabilisce se si tratta del cibo o di uno spazio; nel primo caso il gioco finisce.
- La linea 300 stampa il carattere equivalente al bruco nella posizione L, C.

Se avete afferrato questi concetti siete praticamente in grado di programmare da soli dei semplici giochi di movimento.

Nello Spectrum il display file è organizzato in modo da essere pressoché illeggibile. Può essere usata la funzione SCREEN$, e lo faremo più avanti. Per ora nella versione per lo Spectrum inseriamo la linea:

```
302 IF y = 1 AND
      x = c THEN GOTO 600
```

(vedi figura 3). Questa forma però non è adatta quando gli oggetti da riconoscere sono molti e disposti a caso.

Ma torniamo al nostro amico Tobia, che generalmente muore di fame senza riuscire a raggiungere il cibo. Ci fa una grande pena e decidiamo di aiutarlo, modificando il suo comportamento. Per fare questo, dobbiamo intervenire sul listato 1, nella parte che determina i movimenti: le linee 370-395.

Tobia si muove lungo le coordinate L, C ed i valori di L, C vengono determinati dalla linea 375. Dobbiamo fare in modo che Tobia non si aggiri più nel prato in modo casuale, ma segua una particolare direzione, cioè muova direttamente verso il cibo: in pratica tenti di raggiungere le coordinate Y, X seguendo il percorso più breve.

```
   1 REM TOBIA BRUCO POCO ASTUTO
   2 REM IL CICLO FOR Z REGOLA L
A VELOCITA
   5 SLOW
 110 LET Y=INT (RND*12)+10
 120 LET X=INT (RND*12)
 130 LET L=0
 140 LET C=25
 150 LET A$="█"
 195 PRINT AT L,C;A$
 220 FOR K=1 TO 500
 225 PRINT AT Y,X;CHR$ 151
 230 PRINT AT L,C;
 240 IF PEEK (PEEK 16398+256*PEE
K 16399)=151 THEN GOTO 600
 300 PRINT A$
 305 FOR Z=1 TO 25
 306 NEXT Z
 310 PRINT AT L,C;" "
 370 LET A=INT (RND*4)
 375 IF RND>.49 THEN LET A=-A
 380 LET L=L+A
 385 LET C=C-A
 390 IF L<0 OR L>21 THEN LET L=1
0
 395 IF C<0 OR C>31 THEN LET C=X
-1
 500 NEXT K
 550 PRINT "PECCATO..TOBIA E MOR
TO DI FAME"
 555 STOP
 600 PRINT AT Y-1,X-1;"GNAM"
 605 PAUSE 100
 606 CLS
 610 RUN
```

Listato 1. *Il carattere CHR$ 151 (CHR$ 42 + INVERSE nello Spectrum) è un asterisco inverso e rappresenta il cibo. Il bruco è rappresentato dal carattere SPAZIO INVERSO (CHR$ 128, nello Spectrum CHR$ 143), e si muove sullo schermo alla ricerca del cibo in modo casuale.*

```
   1 REM TOBIA SI ABBUFFA

 340 LET L=L-(Y<L)+(Y>L)
 350 LET C=C-(X<C)+(X>C)
 360 PRINT AT Y,X;" "

 380 LET Y=Y+A
 385 LET X=X-A
 390 IF Y<0 OR Y>21 THEN LET Y=1
0
 392 IF X<0 OR X>31 THEN LET X=1
0
```

Listato 2. *La "routine intelligente" delle linee 340 e 350 fa sì che il bruco raggiunga il cibo seguendo la via più breve.*

Battiamo sul listato 1 già caricato le linee del listato 2, e diamo il RUN. Vedremo Tobia precipitarsi sul cibo e farne grosse scorpacciate: il cibo sarà talmente spaventato che cercherà persino di fuggire, ma inevitabilmente finirà per essere raggiunto dal voracissimo bruco.

Evidentemente, con le modifiche apportate dal listato 2, facciamo in modo che le coordinate L, C su cui viaggia Tobia diventino nel più breve tempo possibile uguali alle coordinate Y, X sulle quali viaggia il ci-

bo. Questo viene ottenuto con le linee:

340 LET L = L − (Y<L) + (Y>L)
350 LET C = C − (X<C) + (X>C)

Analizziamo nel dettaglio lo svolgimento di queste due linee, immaginando una situazione iniziale in cui Tobia si trova alle coordinate:

L, C = 8, 9

ed il cibo rimane fermo alle coordinate:

Y, X = 11, 7

Le linee 340 e 350 vengono eseguite tante volte quanto è il valore del ciclo FOR K (linee 220-500) cioè 500 volte, a meno che non accada prima che L diventi uguale a Y e C diventi uguale a X nello stesso momento. Come abbiamo visto, la linea 240 non verifica che L, C e Y, X siano uguali, controlla invece se la posizione L, C dello schermo contiene il carattere CHR$ 151 che rappresenta il cibo.

In questo caso, evidentemente la posizione L, C è uguale a Y, X perché il cibo è situato proprio in Y, X. Torniamo al ciclo FOR K e vediamo cosa accade alle linee 340 e 350 nel corso delle varie esecuzioni.

K = 1 L, C = 8, 9
Y, X = 11, 7

Sostituendo a L, C e Y, X i rispettivi valori, otteniamo:

340 LET L=8−(11<8)+(11>8)
350 LET C=9−(7<9)+(7>9)

L'espressione (11<8) della linea 340 vale 0, perché è falsa in quanto 11 non è minore di 8. Viceversa l'espressione (11>8) è vera perchè 11 è effettivamente maggiore di 8, quindi vale 1. Di conseguenza la linea 340 va interpretata:

L = L − 0 + 1

e poiché L valeva 8, avremo che:

L = 8 − 0 + 1 = 9

Nella linea 350 succede che:

- l'espressione (7<9) vale 1
- l'espressione (7>9) vale 0

quindi la linea 350 va letta così:

C = C − 1 + 0

e poiché C valeva 9, avremo che:

C = 9 − 1 + 0 = 8

Manteniamo invariati i valori di Y, X per semplificare le cose, e passiamo alla seconda esecuzione del ciclo FOR K:

K = 2 L, C = 9, 8
Y, X = 11, 7

Le linee 340 e 350 vanno interpretate così:

340 LET L=L − (11<9)+(11>9)
350 LET C=C − (7<8)+(7>8)

cioè:

340 LET L = 9 − 0 + 1 = 10
350 LET C = 8 − 1 + 0 = 7

Alla terza esecuzione del ciclo FOR K abbiamo i seguenti valori:

K = 3 L, C = 10, 7
Y, X = 11, 7

Le linee 340 e 350 si leggono così:

340 LET L=L − (11<10) + (11>10)
350 LET C=C − (7<7) + (7>7)

cioè:

L = 10 − 0 + 1 = 11
C = 7 − 0 + 0 = 7

Tobia è arrivato sulle coordinate 11, 7, cioè sul cibo, con tre soli spostamenti, seguendo la via più breve.

```
   1 REM ICBM ATTAK
   2 REM IL CICLO FOR U (375) RE
GOLA LA VELOCITA
   5 SLOW
  40 GOTO 70
  50 PRINT AT 19,0;"
"
  52 PRINT TAB 10;"ICBM ATTAK"
  54 PRINT "
"
  60 RETURN
  70 FOR K=12 TO 18
  72 LET JK=INT (RND*5)+1
  74 FOR W=16-JK TO 16
  76 PRINT AT W,K;CHR$ 8
  78 IF RND>.75 THEN PRINT AT W,
K;CHR$ 130
  80 NEXT W
  82 NEXT K
  85 LET P=0
  86 LET JK=0
  87 LET N=0
  88 LET O=0
 100 GOSUB 50
 102 LET D=18
 103 LET E=16
 105 LET L=10
 106 LET C=15
 200 DIM Y(3)
 205 DIM X(3)
 210 LET Y(1)=INT (RND*18)+1
 211 LET X(1)=INT (RND*7)
 212 LET Y(2)=0
 213 LET X(2)=INT (RND*27)+5
 214 LET Y(3)=INT (RND*10)+9
 215 LET X(3)=INT (RND*7)+25
 300 FOR Z=1 TO 500
 302 IF INKEY$<>"" THEN LET JK=1
 303 IF JK=1 THEN GOTO 370
 350 LET K=3
```

Listato 3. *Tre missili contemporaneamente si dirigono verso il centro della città. Sta a te intercettarli e distruggerli. Premi i tasti direzionali (5, 6, 7 e 8) per muoverti sullo schermo.*
Il carattere CHR$ 138 (linea 50) si ottiene con Graphics e F. CHR$ 130 si ottiene con Graphics e W. CHR$ 128 è lo spazio nero (Graphics e spazio) e CHR$ 151 è l'asterisco inverso (Graphics e B). La routine che fa convergere i missili sulla città (linee 410-420) è uguale a quella che dirige il bruco sul cibo (linee 340 e 350 nel listato 2).

```
 355 PRINT AT Y(K),X(K);CHR$ 151
 360 IF Y(K)=D AND X(K)=E THEN G
OTO 1500
 365 LET K=K-1
 368 IF K<>0 THEN GOTO 355
 370 PRINT AT N,O;" "
 371 PRINT AT L,C;
 372 IF PEEK (PEEK 16398+256*PEE
K 16399)=151 THEN GOTO 1000
 373 PRINT AT L,C;"*"
 374 IF JK=1 THEN GOTO 377
 375 FOR U=1 TO 25
 376 NEXT U
 377 LET N=L
 378 LET O=C
 380 LET C=C+(INKEY$="8")-(INKEY
$="5")
 385 LET L=L+(INKEY$="6")-(INKEY
$="7")
 390 IF JK=1 THEN GOTO 499
 400 LET K=3
 405 PRINT AT Y(K),X(K);" "
 410 LET Y(K)=Y(K)-(D<Y(K))+(D>Y
(K))
 420 LET X(K)=X(K)-(E<X(K))+(E>X
(K))
 430 LET K=K-1
 440 IF K<>0 THEN GOTO 405
 499 LET JK=0
 500 NEXT Z
1000 LET V=3
1005 PRINT AT Y(V),X(V);" "
1010 LET V=V-1
1015 IF V<>0 THEN GOTO 1005
1020 LET P=P+(10-(L-D))**2
1030 PRINT AT 21,16-(LEN STR$ P)
/2;P
1040 GOTO 102
1500 PAUSE 100
1510 CLS
1520 RUN
```

Il fatto che il cibo tenti di fuggire è del tutto irrilevante, perché la fuga si svolge secondo criteri di casualità mentre invece l'inseguimento e la cattura vengono effettuati con fredda e calcolata determinazione.

Immagino che a questo punto sarà sorto tra i lettori un vasto movimento pro-cibo. Potrei suggerire di rendere intelligente anche la fuga, con queste variazioni al listato:

```
380 LET Y=Y + (L<Y) - (L>Y)
385 LET X=X + (C<X) - (C>X)
```

A conclusione di questo paragrafo osserviamo che abbiamo usato una *routine intelligente* (la linea 340 e la 350 sono due versioni di una identica routine) per rendere efficace la ricerca del cibo, e potremmo usare una seconda routine intelligente (derivata dalla prima) per rendere il cibo imprendibile. In questo secondo caso però peggioreremmo la situazione iniziale, quando almeno qualche volta Tobia mangiava. Perciò continueremo ad usare la prima routine lasciando che il cibo segua il suo destino naturale, che consiste nell'essere predato.

Due programmi

La routine intelligente appena presentata può essere usata vantaggiosamente in una miriade di occasioni diverse. Nei listati 3 e 4 si determina il movimento degli aggressori.

Nel listato 3 (ICBM attak) voi siete l'asterisco che appare più o meno sopra la città da difendere. Da tre opposte direzioni 3 missili ICBM piombano sulla città cercando di distruggerla. Il vostro compito è di contrastare il loro attacco, intercettandone almeno uno. I missili viaggiano sulle coordinate Y, X; siccome abbiamo 3 missili, conserviamo le coordinate di ciascuno di essi negli elementi dei due vettori Y (3) e X (3). Le linee 410 e 420 fanno in modo che i missili si dirigano verso il centro della città (coordinate D, E) modificando i valori di Y (K), X (K) appunto in funzione di D, E.

Potete muovere premendo 5, 6, 7 e 8. Ogni volta che un missile viene intercettato il punteggio si incrementa e tutti i missili riprendono l'attacco dalle posizioni iniziali.

La linea 372 interroga il display file per sapere se la posizione L, C che sta per essere occupata dal vostro intercettore contiene un missile (CHR$ 151, asterisco inverso). In caso affermativo il programma passa alla linea 1000.

Le linee 380 e 385 vanno interpretate tenendo conto che se le affermazioni tra parentesi sono vere valgono 1, se sono false valgono zero. Quindi, se voi avete premuto per esempio 8, la linea 380 vale:

C + 1 − 0

e l'intercettore si sposta verso destra di una posizione perché C rappresenta la coordinata relativa alle colonne.

Se invece avete premuto 6, allora:

C = C + 0 − 0
L = L + 1 − 0

cioè C rimane invariata, e L si incrementa di 1 (l'intercettore si abbassa di una posizione).

Nel listato 4 (*Ardito convoglio*) siete alla guida di un convoglio navale ed avete quasi raggiunto il porto di destinazione (le tre H inverse sulla destra) quando un nugolo di motovedette nemiche, armate fino ai denti,

```
   1 REM ARDITO CONVOGLIO
   2 REM IL CICLO FOR U (280) RE
GOLA LA VELOCITA
   5 SLOW
  50 GOTO 70
  55 CLS
  56 PRINT "▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒"
  60 PRINT TAB 5;"ARDITO CONVOGL
IO"
  62 PRINT "████████████████████████████████"
  65 RETURN
  70 PRINT AT 19,0;
  72 GOSUB 56
  73 LET N=0
  74 LET O=0
  78 PRINT AT 13,29;"███";TAB 29
;"▛▛▛";TAB 31;"▛";TAB 31;"▛";TAB
 31;"▛";TAB 31;"▛"
  90 FOR K=3 TO 17
  92 FOR W=6 TO 27
  94 IF RND>.92 THEN PRINT AT K,
W;"O"
  96 NEXT W
  97 NEXT K
  98 LET P=0
 100 LET L=13
 102 LET C=0
 104 PRINT AT L,C;CHR$ 151
 110 DIM Y(3)
 112 DIM X(3)
 114 LET Y(1)=INT (RND*5)
 115 LET X(1)=INT (RND*11)
 116 LET Y(2)=0
 118 LET X(2)=INT (RND*16)+11
 119 LET Y(3)=INT (RND*12)
 120 LET X(3)=31
 200 FOR J=1 TO 500
 210 LET Z=1
 215 PRINT AT Y(Z),X(Z);
 220 IF PEEK (PEEK 16398+256*PEE
K 16399)=52 THEN GOSUB 1000
 222 IF Y(Z)=L AND X(Z)=C THEN G
OTO 1500
 225 PRINT AT Y(Z),X(Z);CHR$ 130
 230 LET Z=Z+1
 235 IF Z<4 THEN GOTO 215
 236 PRINT AT N,O;" "
 250 PRINT AT L,C;
 255 IF PEEK (PEEK 16398+256*PEE
K 16399)=52 THEN GOTO 1500
 260 IF PEEK (PEEK 16398+256*PEE
K 16399)=138 THEN GOTO 2000
 270 PRINT AT L,C;CHR$ 151
 280 FOR U=1 TO 25
 282 NEXT U
 285 LET N=L
 290 LET O=C
 300 LET C=C+(INKEY$="8")-(INKEY
$="5")
 305 LET L=L+(INKEY$="6")-(INKEY
$="7")
 350 LET Z=1
 355 PRINT AT Y(Z),X(Z);" "
 360 LET Y(Z)=Y(Z)-(L<Y(Z))+(L>Y
(Z))
 370 LET X(Z)=X(Z)-(C<X(Z))+(C>X
(Z))
 380 LET Z=Z+1
 385 IF Z<4 THEN GOTO 355
 500 NEXT J
 510 RUN
1000 PRINT AT Y(Z),X(Z);"O"
1002 LET Y(Z)=INT (RND*10)
1004 LET X(Z)=31
1010 LET P=P+INT ((RND*8)+1)**2
1020 PRINT AT 21,16-(LEN STR$ P)
/2;P
1030 RETURN
1500 GOSUB 55
1510 PRINT AT 10,0;"PECCATO.."
1512 PRINT "IL CONVOGLIO E STATO
 DISTRUTTO"
1520 GOTO 2500
2000 GOSUB 55
2010 PRINT AT 10,0;"O.K. MISSION
E RIUSCITA"
2020 PRINT AT 14,16-(LEN STR$ P)
/2;P
2500 INPUT U$
2510 CLS
2520 RUN
```

Listato 4. *Sei alla guida di un convoglio navale e devi raggiungere il porto evitando le motovedette nemiche. Queste puntano su di te (linee 360 e 370) e l'unico modo per evitarle consiste nel ripararsi dietro gli scogli sommersi. Il carattere CHR$ 8 si ottiene con Graphics e H, CHR$ 6 con Graphics e T. I due asterischi della linea 1010 indicano l'elevazione a potenza e si ottengono con SHIFT e H.*

vi piomba addosso. Fortunatamente per voi la zona è piena di scogli sommersi di cui conoscete la posizione: l'unica tattica di salvezza consiste nell'attirare verso di voi le motovedette e ripararvi all'ultimo momento dietro gli scogli, cosicché, nello slancio preso per abbordarvi, i natanti nemici vi si infrangono addosso. Avanzando e riparandovi di scoglio in scoglio potete forse raggiungere la sicurezza del porto. Attenzione, gli scogli sono pericolosi anche per voi. Anche in questo gioco potete muovere premendo 5, 6, 7 e 8.

Tobia e gli ostacoli

Torniamo al nostro amico Tobia, che nel frattempo si è rimpinzato ed è cresciuto veramente molto. Caricate il listato 5 e date il RUN: lo vedrete muoversi sullo schermo e strisciare con un andamento tipico in direzione del cibo. Stavolta però si verifica un inconveniente del tutto imprevisto: davanti al cibo si erge un ostacolo, un lungo muro. Una lucertola, o un topo non ci penserebbero due volte ad aggirarlo, ma Tobia ha un solo milligrammo di cervello per cui continua a puntare nella direzione in cui sa trovarsi l'agognato cibo, anche se non riesce a fare un passo avanti perché il muro è insuperabile frontalmente.

Ad un certo punto accade che il cibo, sentendosi sicuro, si fa più intraprendente e si affaccia da un lato del muro per sbeffeggiare Tobia. Mal gliene incoglie: un secondo bruco sbuca fuori da chissà dove e se lo mangia. Tobia però continua a rimanere a bocca asciutta. Volete aiutarlo ad aggirare il muro?

Registrate a parte il listato 3 che servirà ancora, e battetegli sopra le linee del listato 4 dando poi il RUN. Vedrete Tobia camminare deciso fino al muro, qui fermarsi e dopo un momento di esitazione puntare direttamente verso il basso, fino alla base del muro; lo vedrete poi risalire lungo il lato opposto fino a catturare il cibo.

Il cammino di Tobia procede regolare finché non incontra di fronte a sé il muro (carattere CHR$ 136). La linea 236 riconosce questa situazione e fa saltare alla linea 700 dove il percorso di Tobia viene modificato in conseguenza. In pratica, si aumenta di uno il valore di L lasciando invariato il valore di C, cosicché Tobia si abbassa di una posizione nello schermo. La linea 760 controlla se di fronte c'è ancora il muro. In questo caso, il programma passa alla linea 360, cioè vengono saltate le linee 335 e 340 che avrebbero indirizzato il bruco verso il cibo, quindi addosso al muro. Quando invece alla linea 760 verifichiamo che il muro è finito, (cioè quando il carattere di fronte a Tobia non è più un carattere CHR$ 136) allora il controllo dei movimenti ritorna alle linee 335 e 340, che fanno puntare direttamente sul cibo.

Anche stavolta dunque tutto bene: abbiamo insegnato al nostro amico bruco a superare gli ostacoli.

Le linee 9250-9340 del listato 5 regolano gli spostamenti del bruco sullo schermo. Il bruco è composto da una testa (spazio inverso) seguita da 4 caratteri O e da un carattere spazio: complessivamente 6 caratte-

ri, che richiedono, per essere stampati, l'indicazione di 6 coppie di coordinate. Queste sono conservate nei vettori P (6) e Q (6). Quando il bruco si muove, imita l'andamento di un treno (dove passa il locomotore passeranno prima o poi tutti i vagoni). Così dove passa la testa del bruco passeranno tutte le quattro O che ne costituiscono il corpo, e per ultimo il carattere spazio che cancella le tracce del passaggio.

Con il ciclo FOR W delle linee 9250/9280 facciamo in modo che gli elementi di P e Q diventino di volta in volta uguali all'elemento che li precede. Quindi l'elemento 6 diventa uguale all'elemento 5, e così via fino all'elemento 2 che diventa uguale all'elemento 1, (la vecchia posizione della testa del bruco). L'elemento 1 diventa naturalmente uguale ad L, C che rappresenta la posizione del locomotore, cioè della testa. Il ciclo FOR W delle linee 9310-9330 stampa il bruco nelle nuove posizioni: ogni anello del suo corpo va ad occupare la posizione dell'anello precedente, e per ultimo il carattere spazio cancella l'ultima O.

Tobia sceglie la strada più corta

Purtroppo nel mondo di Tobia la concorrenzialità è assai alta; abbiamo notato come, di fronte alle esitazioni del nostro amico, un secondo bruco sbucava fuori e si impadroniva della sua preda.

Le possibilità di sopravvivenza saranno tanto maggiori, quanto più privo di incertezze sarà il suo comportamento. Se capita di fronte al muro nella posizione più alta, non sarebbe logico superarlo puntando verso l'alto anziché aggirarlo tutto verso il basso?

Scegliendo la strada più corta accorcia i tempi e può evitare che il cibo gli venga rubato da altri bruchi.

Allora ricaricate il listato 5, e su questo battete il listato 7. Dando il RUN, vedrete una serie di ostacoli di fronte a Tobia; il nostro amico non si perderà d'animo e li supererà tutti, scegliendo sempre la via più conveniente cioè la più breve.

In pratica, quando Tobia arriva di fronte al muro, il computer inizializ-

```
   1 REM TOBIA INCONTRA UN OSTAC
OLO
   2 REM IL CICLO FOR Z (300)REG
OLA LA VELOCITA
   5 SLOW
 100 DIM P(6)
 102 DIM Q(6)
 110 LET Y=10
 120 LET X=4
 130 LET L=0
 140 LET C=25
 150 LET A$=CHR$ 128+CHR$ 52+CHR
$ 52+CHR$ 52+CHR$ 52+CHR$ 0
 160 FOR K=1 TO 6
 170 LET P(K)=L
 180 LET Q(K)=25+K
 190 NEXT K
 195 PRINT AT L,C;A$
 200 FOR K=4 TO 18
 202 PRINT AT K,8;CHR$ 136
 204 NEXT K
 220 FOR K=1 TO 500
 225 PRINT AT Y,X;"■"
 230 PRINT AT L,C;
 235 IF PEEK (PEEK 16398+256*PEE
K 16399)=151 THEN GOTO 600
 240 GOSUB 9250
 300 FOR Z=1 TO 25
 305 NEXT Z
 306 IF PEEK (PEEK 16398+256*PEE
K 16399)=136 THEN GOTO 340
 310 GOSUB 9310
 335 LET C=C-(X<C)+(X>C)
 340 LET L=L-(Y<L)+(Y>L)
 360 PRINT AT Y,X;" "
 370 LET A=RND+RND
 375 IF RND>.49 THEN LET A=-A
 380 LET Y=Y+INT A
 390 IF Y<0 OR Y>21 THEN LET Y=1
0
 500 NEXT K
 600 PRINT AT Y-1,X-1;"GNAM"
 605 PAUSE 100
 606 CLS
 610 RUN
9250 FOR W=6 TO 2 STEP -1
9260 LET P(W)=P(W-1)
9270 LET Q(W)=Q(W-1)
9280 NEXT W
9290 LET P(1)=L
9300 LET Q(1)=C
9305 RETURN
9310 FOR W=6 TO 1 STEP -1
9320 PRINT AT P(W),Q(W);A$(W)
9330 NEXT W
9340 RETURN
```

Listato 5. *In questo programma il bruco viene rappresentato in modo completo (stringa A$) e si muove con andamento sinuoso sullo schermo. Le linee dalla 9250 alla 9430 determinano il suo movimento. Il bruco non riesce a superare l'ostacolo che lo divide dal cibo.*

```
   1 REM TOBIA SUPERA GLI OSTACO
LI
   2 REM IL CICLO FOR Z (9305) R
EGOLA LA VELOCITA
 236 IF PEEK (PEEK 16398+256*PEE
K 16399)=136 THEN GOTO 700
 700 LET C=C+1
 720 GOSUB 9250
 730 LET L=L+1
 740 LET C=C-1
 750 PRINT AT L,C;
 760 IF PEEK (PEEK 16398+256*PEE
K 16399)=136 THEN GOTO 360
 770 GOTO 335
9305 FOR Z=1 TO 25
9306 NEXT Z
```

Listato 6. *Battendo le linee del listato 4 sul listato 3, e cancellando le linee 300, 305, 306, 310, il bruco acquista la capacità di superare gli eventuali ostacoli che lo dividono dal cibo.*

```
   1 REM TOBIA SCEGLIE LA STRADA
 PIU CORTA
   2 REM IL CICLO FOR U (9305) R
EGOLA LA VELOCITA
   5 SLOW
  90 LET JK=0
 120 LET X=2
 130 LET L=12
 204 PRINT AT K-2,5;CHR$ 136
 206 PRINT AT K+2,16;CHR$ 136
 208 PRINT AT K-2,20;CHR$ 136
 215 NEXT K
 235 IF PEEK (PEEK 16398+256*PEE
 236 IF PEEK (PEEK 16398+256*PEE
K 16399)=136 THEN GOTO 700
 336 LET JK=0
 700 IF JK<>0 THEN GOTO 750
 701 LET SU=0
 702 LET GIU=0
 710 FOR Z=PEEK 16398+256*PEEK 1
6399 TO PEEK 16396+256*PEEK 1639
7 STEP -33
 720 IF PEEK Z=151 OR PEEK Z=0 T
HEN GOTO 730
 722 LET SU=SU+1
 725 NEXT Z
 730 FOR Z=PEEK 16398+256*PEEK 1
6399 TO PEEK 16396+256*PEEK 1639
7+726 STEP 33
 735 IF PEEK Z=151 OR PEEK Z=0 T
HEN GOTO 746
 740 LET GIU=GIU+1
 745 NEXT Z
 746 LET JK=1
 750 LET C=C+1
 752 GOSUB 9250
 755 LET C=C-1
 760 LET L=L-(GIU>=SU)+(SU>GIU)
 765 PRINT AT L,C;
 770 IF PEEK (PEEK 16398+256*PEE
K 16399)=136 THEN GOTO 360
 775 GOTO 335
9305 FOR U=1 TO 25
9306 NEXT U
```

Listato 7. *Battere le linee del listato 5 sul listato 3 e cancellare le linee 204, 300, 305, 306, 310. Dando il RUN, vedrete che il bruco ha acquistato la capacità di superare gli ostacoli scegliendo la strada più corta.*
I due cicli FOR Z delle linee 710 e 730 misurano la lunghezza del muro leggendo il display file. Il risultato avrebbe potuto essere ottenuto in modo più semplice (vedi la versione per lo Spectrum), ed il metodo presentato in questo listato rappresenta una curiosità.

za due variabili, SU e GIÙ. Conta poi quant'è lungo il muro verso l'alto (SU) e quanto verso il basso (GIÙ) e con la linea 760 decide la direzione verso cui muovere. La linea 760 contiene una routine che funziona in modo analogo a quella della linea 340 nel listato 1.

Infatti quando GIÙ è maggiore di SU, significa che la strada più breve è quella verso l'alto, e allora l'espressione (GIÙ > = SU) vale 1 ed il valore di L viene diminuito di 1, cioè Tobia si sposta verso l'alto: viceversa nel caso opposto.

I due cicli FOR Z (linee 710 e 730) contano la lunghezza del muro verso l'alto e verso il basso in un modo abbastanza singolare, che diventa chiaro leggendo il riquadro sul display file.

Nella versione per lo Spectrum usiamo più semplicemente la versione:

```
710 FOR Z = L TO 0 STEP-1
730 FOR Z = L TO 21
```

Il risultato è lo stesso ma presentando le due versioni credo di offrire maggiori spunti ai lettori.

Facciamoci ora alcune domande. Cosa accade se sistemiamo il cibo all'interno di un labirinto?

Possiamo insegnare a Tobia ad entrarvi, ed a raggiungere il cibo? Ed una volta trovatolo, può uscire dal labirinto seguendo la strada più corta?

Questo è certamente possibile. Tobia deve affrontare il labirinto con una tecnica del tutto particolare, memorizzando tutti i percorsi seguiti all'andata e scegliendo quindi i più brevi per il ritorno. Cioè deve comportarsi come faremmo anche noi entrando in un edificio per la prima volta: in modo intelligente.

Qualcuno obietterà che la sua intelligenza dipende solo dalla complessità della routine che determina il suo comportamento.

In effetti anche la nostra intelligenza dipende spesso dal grado di conoscenza o di ignoranza che abbiamo di un determinato problema. È evidente però che mentre noi possiamo programmare il comportamento di Tobia, né Tobia né il computer possono programmare il nostro. Riprenderemo nel prossimo numero il discorso sul bruco nel labirinto.

Il processo di apprendimento

Il listato 8 contiene la conversione del programma *Master Mind* apparso sul numero 1 di **Personal Software**.

Quando fa la prima mossa, il computer la sceglie a caso tra tutte le mosse possibili. In base alle valutazioni ricevute in risposta (tanti numeri giusti al posto giusto, tanti giusti ma fuori posto) restringe il campo delle mosse possibili.

L'articolo che accompagnava il programma nella versione originale illustrava la tecnica di IA usata dal computer, e vi consiglio di rileggerlo. La tecnica usata è detta del processo di apprendimento, ed è illustrata negli schemi della figura 1, di cui parleremo diffusamente più avanti.

Ovviamente la tecnica del processo di apprendimento costringe il

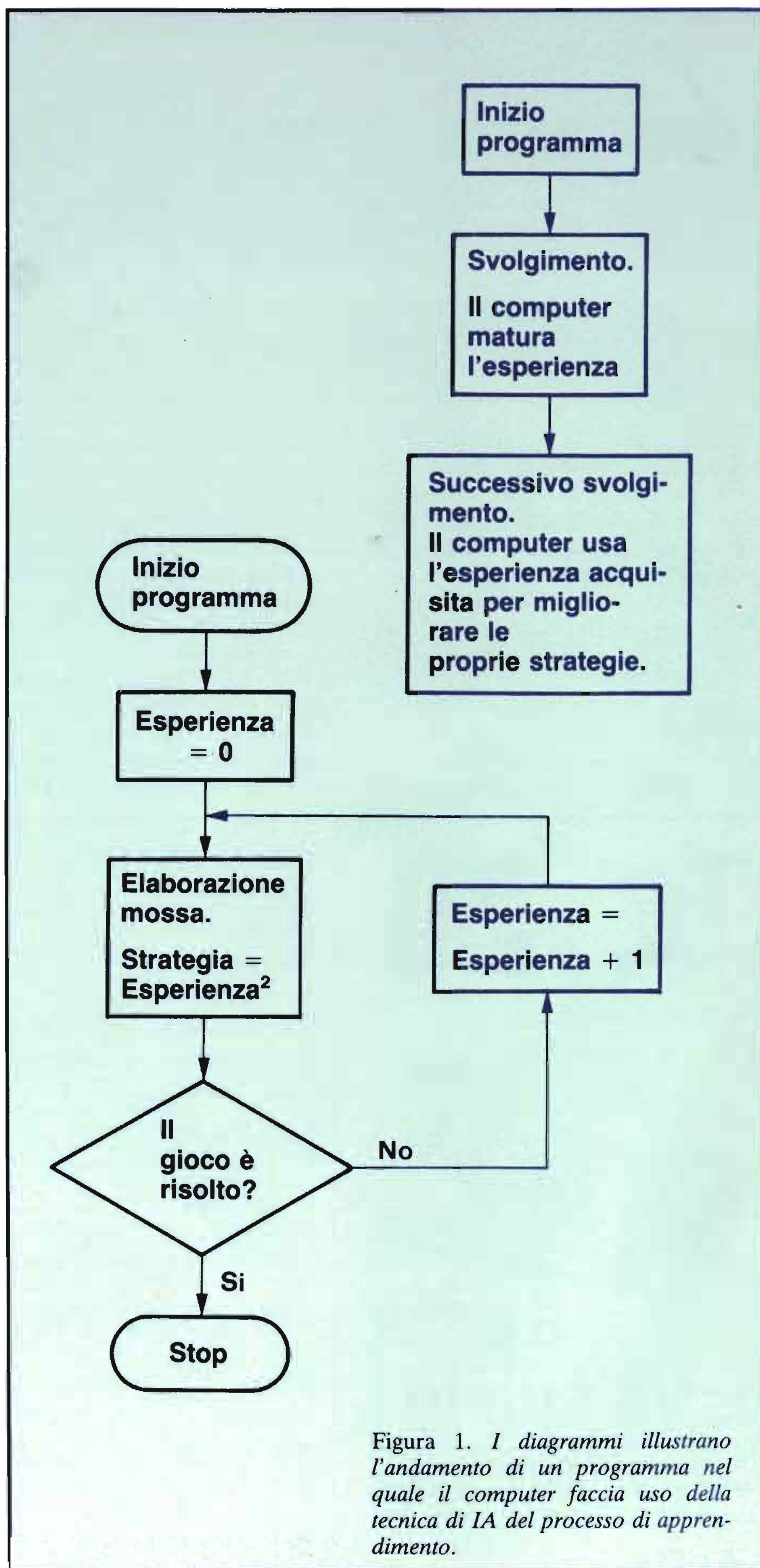


Figura 1. *I diagrammi illustrano l'andamento di un programma nel quale il computer faccia uso della tecnica di IA del processo di apprendimento.*

computer a ragionare veramente sulle mosse che fa, e se avete dei dubbi in proposito paragonate l'andamento dell'algoritmo che determina la mossa del computer con il ragionamento che fareste voi per decidere quella mossa.

Per esempio, se introducete il numero:

4455

e ricevete in risposta la seguente valutazione:

0 NERI (nessun numero giusto)
0 BIANCHI (nessuno giusto fuori posto)

stabilite con sicurezza che i numeri 4 e 5 non sono presenti nel numero segreto, e decidete di non adoperare più il 4 ed il 5 nella formulazione dei prossimi tentativi. Il computer prende la stessa decisione. Se poi introducete il numero:

2233

e ricevete come valutazione 2 neri e 2 bianchi, oltre a non adoperare più né il 4 né il 5 decidete di non adoperare neppure l'1 e il 6. Il computer fa altrettanto: con la tecnica del processo di apprendimento restringe il campo delle mosse possibili fino a limitarlo ad una sola, che ovviamente sarà quella esatta.

Non tutti i programmi intelligenti usano le tecniche del processo di apprendimento. Un esempio è costituito dal programma *Pagliette (Conversione di programmi* per ZX81, seconda parte) pubblicato in questo stesso numero. Il computer gioca molto bene, ma sempre nello stesso modo perché tutta la sua strategia si basa sulla ricerca di particolari posizioni sicure: egli non migliora il livello di gioco con l'esperienza e vince solo se le condizioni gli sono favorevoli, cioè se il numero di pagliette iniziale è propizio e soprattutto se l'avversario non conosce la sua strategia (in questo caso è vera l'affermazione che l'intelligenza artificiale è una funzione della ingenuità umana).

A volte in programmi come questo la routine che determina la mossa può essere talmente ben congegnata da rendere il computer praticamente imbattibile quando le circostanze sono favorevoli.

È il caso del programma *Almeno una volta* (listato 9).

In questo gioco perde chi è costretto a togliere l'ultimo cubetto. Caricate il programma e provate a vincere almeno una volta, se ci riuscite. Non solo il computer vi batterà senza fatica, ma troverà anche il tempo, mentre vince, di far volare stormi di uccelli sullo sfondo di placidi panorami agresti.

Se qualche vostro amico è scettico sulle capacità del computer, sottoponetegli questo programma e gli farete cambiare idea.

Apparentemente il giocatore ha tutti i vantaggi: muove per primo e può togliere da 1 a 4 cubetti. Ne può anche aggiungere da 1 a 4, se i cubetti sono più di 10 e meno di 31. Eppure la vittoria va sempre al computer. Cerchiamo di capire il perché.

In giochi come questo (per esempio il già citato *Pagliette*, e in tutti i giochi della famiglia del NIM) esi-

```
   1 REM ALMENO UNA VOLTA
   2 REM **********************
 100 GOSUB 9936
 112 LET VI=0
 114 LET VT=0
 116 LET CP=1
 118 LET BMD=0
 122 LET AG=0
 124 LET SC=0
 200 GOSUB 9000
 210 PRINT AT 10,0;"COME TI CHIA
MI ?"
 215 INPUT G$
 220 IF LEN G$>10 THEN LET G$=G$
(10 TO )
 225 PRINT G$
 230 PRINT
 240 PRINT "VUOI VEDERE LE REGOL
E? (S/N)"
 245 PAUSE 20000
 250 IF CODE INKEY$=56 THEN GOSU
B 8000
 255 SLOW
 300 LET CM=0
 305 LET TG=0
 310 LET JK=0
 320 LET NUM=26
 360 GOSUB 7000
 500 LET CM=CM+1
 501 LET TG=0
 502 GOSUB 7700
 504 PRINT AT 2,9;CM;AT 3,12;VT
 506 IF CM=1 THEN PRINT AT 18,15
;"TOCCA A TE"
 510 PRINT AT 19,15;"1 PER AGGIU
NGERE";AT 20,15;"2 PER TOGLIERE"
 520 IF INKEY$="" THEN GOSUB 950
0
 530 LET X$=INKEY$
 535 IF CODE X$<>29 AND CODE X$<
>30 THEN GOTO 560
 550 LET SC=VAL X$
 555 IF SC=1 AND NUM<=10 OR SC=1
 AND NUM>30 THEN GOTO 560
 556 GOTO 600
 560 LET BMD=510
 562 GOTO 8200
 600 GOSUB 7500
 610 PRINT AT 18,16;"QUANTI NE";
TAB 16;("TOGLI ?" AND SC=2)+("AG
GIUNGI ?" AND SC=1)
 620 IF INKEY$="" THEN GOSUB 950
0
 630 LET X$=INKEY$
 640 IF CODE X$<29 OR CODE X$>32
 THEN GOTO 680
 650 LET MN=VAL X$
 660 IF SC=2 THEN LET MN=-MN
 670 LET NUM=NUM+MN
 675 IF NUM<1 OR NUM>31 THEN GOT
O 680
 677 GOTO 700
 680 LET BMD=620
 682 GOTO 8200
 700 GOSUB 1000
 702 GOSUB 7500
 705 PRINT AT 18,16;"NE HAI ";AT
 19,16;("TOLTI " AND SC=2)+("AGG
IUNTI " AND SC=1)+X$
 706 IF NUM=1 THEN GOTO 4000
 710 GOSUB 7600
 715 PAUSE 50
 720 PRINT AT 18,1;"TOCCA A ME";
TAB 1;"    "
 730 PAUSE 35
 740 GOSUB 2000
 750 GOSUB 7600
 755 GOSUB 7500
 760 PRINT AT 18,15;("TOLTI " AN
D TG<>0)+("AGGIUN." AND AG<>0)+"
IO: ";TG+AG;"   "
 765 IF NUM=1 THEN GOTO 4200
 770 GOTO 500
 999 STOP
1000 REM STAMPA SCACCHIERA
1010 LET L=INT (NUM/6)*2+6
1020 LET C=16+(NUM-((INT (NUM/6)
)*6))*2
1030 IF JK=1 AND SC=2 THEN GOTO
1500
1040 IF TG<>0 THEN GOTO 1500
1050 FOR K=L TO 6 STEP -2
1060 FOR W=C TO 16 STEP -2
1070 PRINT AT K,W;CHR$ 128
1090 NEXT W
1095 LET C=28
1100 NEXT K
1110 LET JK=1
1120 RETURN
1500 LET KJ=18
1510 FOR K=L+2 TO L STEP -2
1520 FOR W=28 TO KJ STEP -2
1530 PRINT AT K,W;" "
1540 NEXT W
1550 LET KJ=C+2
1560 NEXT K
1580 GOTO 1120
1999 REM IL COMPUTER SCEGLIE LA
MOSSA
2000 LET TG=0
2010 LET AG=0
2012 LET SC=0
2015 IF NUM<=26 THEN GOTO 2030
2018 IF RND>.60 AND NUM<31 THEN
GOTO 2025
2020 LET TG=NUM-26
2022 GOTO 2130
2025 LET AG=31-NUM
2027 GOTO 2130
2030 IF NUM<=5 THEN LET TG=NUM-1
2040 IF TG<>0 THEN GOTO 2130
2050 FOR Z=6 TO NUM STEP 5
2060 FOR W=Z TO Z+4
2070 IF NUM=W THEN GOTO 2090
2080 NEXT W
2085 NEXT Z
2090 IF NUM>10 AND RND>.65 THEN
GOTO 2120
2100 LET TG=W-Z
```

Listato 9. *Il computer visualizza sulla destra dello schermo 26 cubetti. Il giocatore ed il computer a turno possono toglierne o aggiungerne da 1 a 4, ma le aggiunte si possono fare solo quando i cubetti sono meno di 31 e più di 10. Chi toglie l'ultimo cubetto perde. Scopo del gioco è riuscire a battere ALMENO UNA VOLTA il computer. L'impresa è più difficle di quanto sembri.*
Il carattere CHR$ 136 (linea 7110) si ottiene con Graphics e A e non va confuso con CHR$ 8 (Graphics e H). Quando non sapete esattamente quale carattere si voglia intendere con una CHR$ e non avete il manuale a portata di mano, date il comando diretto: PRINT CHR$ x e cercate poi sulla tastiera il carattere che vedrete apparire sullo schermo.

stono delle posizioni sicure: cioè, se ad un certo momento del gioco uno dei giocatori riesce a lasciare un certo numero di cubetti sul campo dopo la sua mossa, ha buone probabilità di vincere; talvolta ne ha la certezza.

La posizione più sicura è ovviamente 1, perché lasciando 1 cubetto l'avversario sarà costretto a toglierlo ed avrà perso.

La seconda posizione sicura in *Almeno una volta* è 6. Quando il giocatore si trova di fronte 6 cubetti, può al limite:

- toglierne 1; ne rimangono 5, e se l'avversario ne toglie 4 ha vinto;
- toglierne 4; ne rimangono 2, e l'avversario ne toglie 1 assicurandosi la vittoria.

Possiamo affermare che se il giocatore spinge l'avversario nella posizione 6, lo ha già spinto inevitabilmente nella posizione 1.

Esiste una posizione precedente alla 6, dalla quale il giocatore precipiti inevitabilmente nella 6? Sicuramente esiste, ed è la 11. Infatti, dalla posizione 11 il giocatore può togliere al massimo 4 cubetti scendendo a 7, e l'avversario ne toglie uno lasciandolo a 6. E se invece di toglierne 4 li aggiunge? Allora sale a 15, e l'avversario aggiungendone 1 lo sistema a 16.

```
2110 GOTO 2130
2120 LET AG=Z+5-W
2130 LET NUM=NUM+AG-TG
2140 GOSUB 1000
2150 RETURN
4000 PRINT AT 10,18;"O.K.";AT 11
,18;"HAI VINTO TU"
4010 LET VT=VT+1
4020 PAUSE 20000
4030 LET CP=CP+1
4040 GOTO 300
4200 PRINT AT 10,18;"HO VINTO IO
"
4210 LET CP=CP+1
4220 LET VI=VI+1
4225 PRINT AT 19,15;"PREMI UN TA
STO";AT 20,15;"PER CONTINUARE"
4230 PAUSE 20000
4240 GOTO 300
7000 REM STAMPA DEL QUADRO
7005 CLS
7010 PRINT "████████████████████
████████"
7015 PRINT AT 21,0;"████████████
████████"
7020 FOR K=1 TO 20
7022 PRINT AT K,0;CHR$ 128;TAB 3
1;CHR$ 128
7025 NEXT K
7030 PRINT AT 4,1;"█████████████
█████████"
7035 PRINT AT 17,1;"████████████
█████████"
7040 FOR K=4 TO 17
7045 PRINT AT K,14;CHR$ 128
7050 NEXT K
7060 PRINT AT 1,14;"█";TAB 14;"█
";TAB 14;"█"
7065 PRINT AT 18,14;"█";TAB 14;"
█";TAB 14;"█"
7075 PRINT AT 1,2;G$;AT 2,2;"MOS
SA";AT 3,2;"VITTORIE"
7100 PRINT AT 1,18;"ALMENO";AT 2
,18;"UNA";AT 3,18;"VOLTA"
7110 PRINT AT 9,4;"█████";TAB 2;
"████████";TAB 2;"████████";TAB
4;"██████";TAB 3;"█ █ █"
7115 PRINT AT 14,5;"█";TAB 6;"█"
;TAB 6;"█"
7120 PRINT AT 10,13;"█";TAB 12;"
██";TAB 11;"███";TAB 12;"██";TAB
13;"█";TAB 12;"█";TAB 13;"█"
7199 GOSUB 1000
7200 RETURN
7500 PRINT AT 18,15;"
"
7505 PRINT AT 19,15;"
"
7506 PRINT AT 20,15;"
"
7510 RETURN
7600 PRINT AT 18,1;"
"
7605 PRINT AT 19,1;"
"
7607 PRINT AT 20,1;"
"
7610 RETURN
7700 PRINT AT 18,2;"ZX81";TAB 2;
"MOSSA ";CM;TAB 2;"VITTORIE ";VI
7710 RETURN
8000 GOSUB 9000
8010 PRINT AT 7,0;"IO SONO IL TU
O AVVERSARIO IN QUESTO GIOCO"
8020 PRINT "A TURNO POSSIAMO TOG
LIERE O AG- GIUNGERE DA 1 A 4 CU
BETTI ALLA TABELLA CHE VEDRAI S
ULLA DESTRA DEL VIDEO"
8030 PRINT
8040 PRINT "LE AGGIUNTE SI POSSO
NO FARE SOLOQUANDO I CUBETTI SON
O PIU DI 10 O MENO DI 31"
8050 PRINT
8060 PRINT "CHI TOGLIE L ULTIMO
PERDE"
8070 PRINT AT 21,0;"PREMI UN TAS
TO"
8100 PAUSE 20000
8110 RETURN
8200 LET DMB=BMD
8205 LET BMD=0
8210 POKE 16418,0
8220 PRINT AT 23,0;"INPUT ERRATO
 RIPETI PREGO"
8230 PAUSE 120
8240 PRINT AT 23,0;"
"
8250 POKE 16418,2
8260 GOTO DMB
9000 CLS
9010 PRINT "████████████████████
██████"
9011 PRINT
9020 PRINT TAB 7;"ALMENO UNA VOL
TA"
9021 PRINT
9030 PRINT "████████████████████
██████"
9040 RETURN
9502 IF INKEY$<>"" THEN RETURN
9553 LET O$=O$(13)+O$
9555 IF INKEY$<>"" THEN RETURN
9556 LET O$=O$( TO 13)
9557 PRINT AT 6,1;O$
9558 IF INKEY$<>"" THEN RETURN
9560 LET P$=P$(13)+P$
9562 IF INKEY$<>"" THEN RETURN
9563 LET P$=P$( TO 13)
9564 PRINT AT 7,1;P$
9565 IF INKEY$<>"" THEN RETURN
9566 LET Q$=Q$(13)+Q$
9570 IF INKEY$<>"" THEN RETURN
9571 LET Q$=Q$( TO 13)
9572 PRINT AT 8,1;Q$
9574 IF INKEY$<>"" THEN RETURN
9580 GOTO 9500
9936 LET O$="     >     >    "
9938 LET P$=" >       >>>>  "
9940 LET Q$=" >       >>    "
9950 RETURN
```

A questo punto avete capito che la sequenza di posizioni vincenti (o perdenti per chi ci cade) è la seguente:

1 - 6 - 11 - 16 - 21 - 26 - 31...

Ora, il gioco comincia proprio con 26 cubetti, e la prima mossa tocca proprio a voi. Quando il gioco non è ancora cominciato voi avete già perso. Potreste sperare di vincere solo se il vostro avversario si distraesse, ma il computer non si distrae e non sbaglia mai. Dunque il computer nel gioco *Almeno una volta* è imbattibile.

Per confortarvi di questa delusione, vi do due suggerimenti.

Il primo è la sequenza vincente per il gioco *Pagliette*:

1 - 5 - 9 - 13 - 17 - 21 - 25 - 29...

In questo gioco il numero di pagliette viene determinato casualmente all'inizio, per cui potete vincere voi anziché il computer quando il numero di pagliette è idoneo e se evitate di distrarvi. Tuttavia capite bene che se anche voi oltre al computer conoscete la strategia delle posizioni vincenti la partita è già finita ancor prima di essere cominciata.

Il secondo suggerimento consiste nelle modifiche da apportare al programma *Almeno una volta*, per dare anche allo sfidante qualche chance di vittoria. Dovete battere sul programma già caricato le seguenti linee:

```
320 LET NUM = INT (RND * 12) + 15
2105 IF TG = 0 THEN LET TG = INT (RND * 4) + 1
2125 IF AG = 5 THEN LET AG = INT ((RND * 4 AND NUM < 28) + (RND * (31-NUM) AND NUM > 27)) + 1
```

Anche in questo caso però l'esito della partita è già determinato e dipende dal numero di cubetti presenti sullo schermo. Vale la pena di bruciare così un programma come *Almeno una volta*?

Sicuramente no. Nella seconda

Listato 1
```
  2 REM LA LINEA 307 REGOLA LA VELOCITÀ
307 PAUSE 30
```
Listato 2
Nessuna variazione
Listato 3
```
  2 REM LA LINEA 305 REGOLA LA VELOCITÀ
305 PAUSE 30
```
Listato 4
```
  2 REM LA LINEA 9307 REGOLA LA VELOCITÀ
9307 PAUSE 30
```
Listato 5
```
  2 REM LA LINEA 9307 REGOLA LA VELOCITÀ
9307 PAUSE 30
```
Listato 6
Cancellare la linea 376
```
  2 REM LA LINEA 375 REGOLA LA VELOCITÀ
375 PAUSE 30
```
Listato 7
```
  2 REM LA LINEA 283 REGOLA LA VELOCITÀ
283 PAUSE 50
```
Listato 8
Nessuna modifica
Listato 9
Cancellare la linea 100, e tutte le linee dalla 9500 in poi.
```
520 PAUSE 20000
620 PAUSE 20000
```

Figura 2. *Variazioni da apportare ai listati per renderli compatibili allo ZX80 nuova ROM 8 Kbyte.*

Listato 1
Cancellare la linea 240
```
 150 LET a$ = CHR$ 143
 226 PRINT AT y, x; INVERSE 1; CHR$ 42
 302 IF y=l AND x=c THEN GOTO 600
```
Listato 2
Nessuna modifica
Listato 3
Cancellare le linee 2, 235, 300, 305
```
 110 LET y = 14
 130 LET l = 10
 150 PRINT CHR$ 143 + CHR$ 79 + CHR$ 79 + CHR$ 79 + CHR$ 79
     + CHR$ 32
 200 FOR k = 3 TO 18
 202 PRINT AT k, 8; CHR$ 133
 225 PRINT AT y, x; INVERSE 1; CHR$ 42
 230 IF l=y AND c=x THEN GOTO 600
 306 IF CODE SCREEN$ (l, c) = 0 THEN GOTO 340
```
Listato 4
```
 236 IF CODE SCREEN$ (l, c) = 0 THEN GOTO 700
 760 IF CODE SCREEN$ (l, c) = 0 THEN GOTO 360
```
Listato 5
```
 110 LET y = INT (RND * 10) + 6
 130 LET l = 10
 204 PRINT AT k - 2, 5; CHR$ 133
 206 PRINT AT k + 2, 16; CHR$ 133
 208 PRINT AT k - 2, 20; CHR$ 133
 236 IF CODE SCREEN$ (l, c) <> 0 THEN GOTO 700
 710 FOR z = l TO 0 STEP - 1
 720 IF CODE SCREEN$ (z, c) <> 0 THEN GOTO 730
 730 FOR z = l TO 21
 735 IF CODE SCREEN$ (z, c) <> 0 THEN GOTO 746
 770 IF CODE SCREEN$ = 0 THEN GOTO 360
```
Listato 6
Cancellare le linee: 87, 88, 370, 371, 372, 374, 377, 378
```
  50 PRINT AT 19,0; "32 caratteri CHR$ 143"
  54 PRINT "32 caratteri CHR$ 143"
  76 PRINT AT w, k; CHR$ 143
  78 IF RND > .75 THEN PRINT AT w, k; CHR$ 137
 355 PRINT AT y (k), x (k); CHR$ 127
 372 IF y (k) = l AND x (k) = c THEN GOTO 1000
 373 PRINT AT l, c; CHR$ 134
2000 REM CHR$ 143 = SPAZIO NERO, GRAPHICS E 8
```
Listato 7
Cancellare le linee 73, 74, 104, 236, 260, 285, 290
```
  56 PRINT "32 caratteri CHR$ 143"
  62 PRINT "32 caratteri CHR$ 143"
  76 PRINT AT 13, 29; INVERSE 1; "HHH"
  78 PRINT AT 14, 29; CHR$ 134 + CHR$ 134 + CHR$ 134
  79 FOR k = 15 TO 21
  80 PRINT AT k, 31; CHR$ 143
  85 NEXT k
  99 LET jk = 0
 202 IF INKEY$ <> "" THEN LET jk = 1
 203 IF jk = 1 THEN GOTO 250
 220 IF y (z) = l AND x (z) = c THEN GOTO 1500
 222 IF CODE SCREEN$ (y (z), x (z)) <> 32 THEN GOSUB 1000
 225 PRINT AT y (z), x (z); INVERSE 1; "0"
 252 IF l = 13 AND c = 29 OR l = 13 AND c = 30 OR l = 13 AND
     c = 31 THEN GOTO 2000
 255 IF CODE SCREEN$ (l, c) <> 32 THEN GOTO 1500
 270 PRINT AT l, c; CHR$ 127
 290 PRINT AT l, c; "spazio"
 310 IF jk = 1 THEN GOTO 499
 499 LET jk = 0
```

parte di questo articolo, oltre ad insegnare a Tobia a muoversi in un labirinto, cambieremo radicalmente la routine che determina la mossa nel programma *Almeno una volta*. Faremo in modo che il computer impari (nel corso delle partite giocate) a memorizzare le mosse migliori utilizzandole poi nel momento opportuno: faremo in modo che il computer impari a giocare.

Osservazioni sui listati 8 e 9

Nel programma *Gara di Master Mind* il computer gioca molto bene ma è relativamente lento: in media impiega fino a 2 minuti per fare la mossa. Se ciò vi spazientisce considerate che programmi simili richiedono tempi anche più lunghi. Comunque potete adoperare il programma "a senso unico", cioè facendo in modo di essere sempre voi a dover indovinare il numero proposto dal compu-

Listato 8

Cancellare le linee 859, 864

```
 280 PRINT TAB 5; "CROCE = TOCCA A TE"; AT 16, 5; "TESTA = TOCCA A ME"
 645 PRINT AT 6, 19; INVERSE 1: "NERI ?"
 648 IF CODE INKEY$ < 48 OR CODE INKEY$ > 52 THEN GOTO 645
 660 PRINT AT 6, 25; INVERSE 1; "BIANCHI"
 786 PRINT # 1; AT 1, 0; "CALMA, STO PENSANDO..."
 848 PRINT # 1; AT 1, 0; "20 spazi"
 860 PRINT AT 22, 0; "AVRESTI DOVUTO BATTERE"; b; ""; b$; "E"; w; " "; w$
 863 PRINT AT 22, 0; "40 spazi"
```

Listato 9

Cancellare: 7060, 7065, 7110, 7115, 7120, 8210, 8250

```
 100 GOSUB 9900
 250 IF CODE INKEY$ = 115 THEN GOSUB 8000
 535 IF CODE x$ <> 49 OR CODE x$ <> 50 THEN GOTO 560
 610 PRINT AT 18, 16; "QUANTI NE"; AT 19, 16; ("TOGLI ?" AND sc = 2) + ("AGGIUNGI ?" AND sc = 1)
 640 IF CODE INKEY$ < 49 OR CODE INKEY$ > 52 THEN GOTO 680
 720 PRINT AT 18, 1; "TOCCA A ME"; AT 19, 1; INVERSE 1; "VADO"
1070 PRINT AT k, w; CHR$ 143
7010 PRINT "32 caratteri CHR$ 143"
7015 PRINT AT 21, 0; "32 caratteri CHR$ 143"
7016 FOR k = 1 TO 20: PRINT AT k, 14; CHR$ 138: NEXT k
7022 PRINT AT k, 0; CHR$ 143; AT k, 31; CHR$ 143
7030 PRINT AT 4, 1; "14 caratteri CHR$ 143 + 16 caratteri CHR$ 131"
7035 PRINT AT 17, 1; "14 caratteri CHR$ 143 + 16 caratteri CHR$ 140"
7045 PRINT AT k, 14; CHR$ 143
7110 FOR k = 4 TO 8: FOR w = 9 TO 12
7112 INK 4: PRINT AT w, k; CHR$ 144: NEXT w: NEXT k
7114 PRINT AT 10, 2; CHR$ 144; AT 11, 2; CHR$ 144 + CHR$ 144;
     AT 13,3; CHR$ 144; AT 13, 8; CHR$ 144; AT 12, 9; CHR$ 144;
     AT 11, 9; CHR$ 144 + CHR$ 144; AT 10, 9; CHR$ 144
7120 PRINT AT 10, 13; CHR$ 144; AT 11, 12; CHR$ 144 + CHR$ 144;
     AT 12, 13; CHR$ 144; AT 13, 12; CHR$ 144 + CHR$ 144
7130 INK 0
7140 PRINT AT 13, 5; CHR$ 133 + CHR$ 133; AT 14, 6; CHR$ 138;
     AT 15, 6; CHR$ 133; AT 16, 6; CHR$ 143
7145 PRINT AT 14, 13; CHR$ 138; AT 15, 13; CHR$ 133; AT 16, 13; CHR$ 138
7700 PRINT AT 18, 2; "SPECTRUM"; AT 19, 2; "MOSSA"; cm; AT 20, 2; "VITTORIE"; vi
8070 PRINT AT 21, 0; INVERSE 1; "PREMI UN TASTO"
8220 PRINT # 1; AT 1, 0; INVERSE 1; "INPUT ERRATO RIPETI PREGO"
8240 PRINT # 1; AT 1, 0; "30 spazi"
9900 REM GENERAZIONE CARATTERI SPECIALI
9910 POKE USR "a", BIN 01010101
9911 POKE USR "a" + 1, BIN 10101010
9912 POKE USR "a" + 2, BIN 01010101
9913 POKE USR "a" + 3, BIN 10101010
9914 POKE USR "a" + 4, BIN 01010101
9915 POKE USR "a" + 5, BIN 10101010
9916 POKE USR "a" + 6, BIN 01010101
9918 POKE USR "a" + 7, BIN 10101010
9920 POKE USR "b", BIN 00000000
9921 POKE USR "b" + 1, BIN 00001100
9922 POKE USR "b" + 2, BIN 00000010
9923 POKE USR "b" + 3, BIN 00000001
9924 POKE USR "b" + 4, BIN 00000010
9925 POKE USR "b" + 5, BIN 00001100
9926 POKE USR "b" + 6, BIN 00110000
9927 POKE USR "b" + 7, BIN 00000000
9936 LET O$ = "4 spazi" + CHR$ 145 + "4 spazi" + CHR$ 145 + "3 spazi"
9938 LET P$ = "1 spazio" + CHR$ 145 + "6 spazi" + CHR$ 145 + CHR$ 145 + CHR$ 145 + CHR$ 145 + "spazio"
9940 LET Q$ = "uno spazio" + CHR$ 145 + "6 spazi" + CHR$ 145 + CHR$ 145 + "3 spazi"
```

Figura 3. *Variazioni da apportare ai listati per renderli compatibili allo Spectrum.*

```
   1 REM GARA DI MASTER MIND
 100 GOSUB 9500
 145 LET MN=0
 146 LET XI=0
 147 LET XT=0
 210 GOSUB 9000
 230 PRINT AT 8,0;"COME TI CHIAM
I?"
 240 INPUT N$
 242 IF LEN N$>10 THEN LET N$=N$
( TO 10)
 244 PRINT N$
 260 PRINT AT 12,0;"LANCIO UNA M
ONETA PER DECIDERE"
 265 PRINT "CHI GIOCA PER PRIMO"
 270 PRINT
 250 PRINT TAB 5;"TESTA = TOCCA
A ME";TAB 5;"CROCE = TOCCA A TE"
 300 PAUSE 100
 302 LET CT=INT (RND*2)
 305 PRINT AT 18,0;
 310 IF CT=0 THEN PRINT "TESTA D
EVO INDOVINARE IL TUO"
 312 IF CT=1 THEN PRINT "CROCE D
EVI INDOVINARE IL MIO"
 314 PRINT "NUMERO  (PREMI UN TA
STO)"
 320 PAUSE 20000
 370 IF CT=0 THEN GOTO 500
 380 GOSUB 960
 385 GOSUB 9500
 390 GOSUB 560
 392 LET MN=MN+1
 394 GOSUB 4000
 396 GOSUB 9500
 400 GOTO 380
 500 GOSUB 560
 505 GOSUB 9500
 510 GOSUB 960
 512 LET MN=MN+1
 514 GOSUB 4000
 516 GOSUB 9500
 520 GOTO 500
 560 REM ROUTINE CODIFICATORE
 570 LET GU=0
 590 GOSUB 9000
 595 PRINT AT 10,0;"SEI IL CODIF
ICATORE"
 600 PRINT "BATTI UN NUMERO DI 4
CIFRE"
 605 INPUT C$
 607 IF LEN C$<>4 THEN GOTO 605
 615 LET M$=C$
 616 GOSUB 1290
 620 IF ER<>0 THEN GOTO 605
 630 GOSUB 9000
 632 PRINT
 634 PRINT "TENTATIVO  NUMERO"
 635 PRINT "                               
          "

 638 GOTO 680
 638 PRINT AT GU+7,11;T$
 640 PRINT AT 7+GU,0;GU
 645 PRINT AT 6,19;"NERI?"
 646 PAUSE 20000
 647 LET X$=INKEY$
 648 IF CODE INKEY$<28 OR CODE I
NKEY$>32 THEN GOTO 645
 649 LET BB=VAL X$
 650 PRINT AT 7+GU,19;BB
 652 IF BB=4 THEN GOTO 3100
 660 PRINT AT 6,25;"BIANCHI"
 665 PAUSE 20000
 668 LET X$=INKEY$
 670 IF CODE X$<28 OR CODE X$>32
THEN GOTO 660
 672 LET WW=VAL X$
 674 PRINT AT 7+GU,25;WW
 676 GOTO 790
 680 LET C1=1
 681 LET C2=1
 682 LET C3=1
 684 LET C4=0
 685 LET GU=GU+1
 700 IF GU<>1 THEN GOTO 710
 702 FOR I=1 TO 4
 703 LET C(I)=INT (RND*6)+1
 704 NEXT I
 710 IF GU<>2 THEN GOTO 720
 712 LET C(1)=C1
 713 LET C(2)=C2
 714 LET C(3)=C3
 715 LET C(4)=C4
 720 IF GU<=2 THEN GOTO 730
 722 FOR I=1 TO 4
 723 LET C(I)=G(GU-1,I)
 724 NEXT I
 730 IF GU>1 THEN GOSUB 1690
 740 LET T$=""
 750 FOR I=1 TO 4
 751 LET Z$=STR$ C(I)
 752 LET T$=T$+Z$(LEN Z$)
 755 NEXT I
 775 GOTO 638
 790 FOR I=1 TO 4
 792 LET G(GU,I)=C(I)
 794 NEXT I
 796 LET G(GU,5)=BB
 798 LET G(GU,6)=WW
 800 LET M$=C$
 802 GOSUB 1290
 810 FOR I=1 TO 4
 812 LET C(I)=T(I)
 814 NEXT I
 820 LET M$=T$
 822 GOSUB 1290
 830 GOSUB 1480
 840 FOR I=1 TO 4
 842 LET T(I)=C(I)
 844 NEXT I
 850 IF BB=B AND WW=W THEN GOTO
880
 853 IF B=1 THEN LET B$="NERO"
 854 IF B<>1 THEN LET B$="NERI"
 857 IF W=1 THEN LET W$="BIANCO"
 858 IF W<>1 THEN LET W$="BIANCH
I"
 859 POKE 16418,0
 860 PRINT AT 22,0;"AVRESTI DOVU
TO BATTERE ";B;" ";B$;" E ";W;"
";W$
 862 PAUSE 200
 863 PRINT AT 22,0;"

 864 POKE 16418,2
 870 LET G(GU,5)=B
 871 LET G(GU,6)=W
 880 IF B=4 THEN GOTO 3100
 890 IF B+W=4 THEN GOSUB 1840
 900 IF B+W=0 THEN GOSUB 1990
 910 IF B=0 AND W<>0 THEN GOSUB
2110
 935 GOTO 685
 950 RETURN
1000 REM ROUTINE DEL DECIFRATORE
1005 GOSUB 9000
1010 PRINT AT 10,0;"STO PREPARAN
DO IL MIO CODICE"
1015 PAUSE 100
1030 GOSUB 9000
1035 PRINT
1036 PRINT "TENTATIVO  NUMERO  N
ERI  BIANCHI"
1040 PRINT "                      
          "

1050 LET NO=0
1060 FOR I=1 TO 4
1062 LET C(I)=INT (RND*6)+1
1066 NEXT I
1068 PRINT AT 21,0;"BATTI UN NUM
ERO DI 4 CIFRE"
1070 INPUT T$
1075 IF LEN T$<>4 THEN GOTO 1070
1080 LET M$=T$
1082 GOSUB 1290
1090 IF ER<>0 THEN GOTO 1070
1095 PRINT AT 21,0;"
1100 GOSUB 1480
1110 LET NO=NO+1
1122 PRINT AT NO+7,0;NO;TAB 11;M
```

ter, cancellando le linee 394 e 514, ed inserendo la linea:

```
571 RETURN
```

La cifra segreta viene formata utilizzando solo numeri da 1 a 6. Il programma *Almeno una volta* richiede un tocco veloce sui tasti quando si forniscono gli input relativi alla mossa, altrimenti il primo input (1 o 2, aggiungi o togli) viene assunto come valido anche per il secondo. Cioè se premete più a lungo del necessario il 2, il computer capisce che volete togliere cubetti e volete toglierne 2. Le numerose linee:

```
IF INKEY$ <> "" THEN RETURN
```

```
$;TAB 19;B;TAB 25;W
1130 IF B<>4 THEN GOTO 1070
1140 GOSUB 9000
1142 PRINT AT 12,0;"HAI INDOVINA
TO"
1144 PRINT
1146 PRINT "IN ";NO;" TENTATIVI"
1148 PRINT
1149 PRINT "COEFFICIENTE DI ABIL
ITA:"
1150 PRINT 1/NO;" (MAX=1)"
1156 PAUSE 20000
1160 RETURN
1290 REM INIZIALIZZA T(I)
1300 LET ER=0
1310 FOR I=1 TO 4
1320 LET T(I)=VAL M$(I)
1330 IF T(I)<1 OR T(I)>6 THEN LE
T ER=1
1340 NEXT I
1350 RETURN
1480 REM VALUTAZIONE DEL TENTATI
VO
1500 LET B=0
1502 LET W=0
1510 FOR I=1 TO 4
1512 LET W(I)=C(I)
1514 NEXT I
1520 FOR I=1 TO 4
1530 IF W(I)<>T(I) THEN GOTO 155
0
1540 LET W(I)=-1
1542 LET T(I)=-2
1544 LET B=B+1
1550 NEXT I
1560 FOR I=1 TO 4
1570 FOR J=1 TO 4
1580 IF T(I)<>W(J) THEN GOTO 160
0
1590 LET W(J)=-1
1592 LET W=W+1
1594 GOTO 1602
1600 NEXT J
1602 NEXT I
1620 RETURN
1690 REM ROUTINE PER NUOVO TENTA
TIVO
1700 GOSUB 2220
1710 FOR L=1 TO GU-1
1720 FOR J=1 TO 4
1730 LET T(J)=G(L,J)
1740 NEXT J
1750 GOSUB 1480
1760 IF G(L,5)=B AND G(L,6)=W TH
EN GOTO 1780
1770 GOTO 1700
1780 NEXT L
1790 RETURN
1870 FOR I=1 TO 6
1872 LET E(I)=I
1874 NEXT I
1880 FOR I=1 TO 4
1890 LET E(G(GU,I))=0
1900 NEXT I
1910 FOR I=1 TO 6
1920 IF E(I)<>0 THEN GOSUB 3200
1930 NEXT I
1940 RETURN
2020 FOR I=1 TO 4
2030 FOR J=1 TO 4
2040 LET D(G(GU,I),J)=0
2050 NEXT J
2052 NEXT I
2060 RETURN
2140 FOR I=1 TO 4
2150 LET D(G(GU,I),I)=0
2160 NEXT I
2170 RETURN
2220 REM ROUTINE PER ESTRARRE UN
 NUMERO DALLA TABELLA DELLE POSS
IBILITA
2230 GOTO 2360
2240 IF D(C1,1)<>0 THEN GOTO 300
0
2250 LET C1=C1+1
2260 IF C1<7 THEN GOTO 2240
2270 GOTO 3020
2280 IF D(C2,2)<>0 THEN GOTO 303
0
2290 LET C2=C2+1
2300 IF C2<7 THEN GOTO 2280
2310 LET C2=1
2312 GOTO 2250
2320 IF D(C3,3)<>0 THEN GOTO 304
0
2330 LET C3=C3+1
2340 IF C3<7 THEN GOTO 2320
2350 LET C3=1
2355 GOTO 2290
2360 LET C4=C4+1
2370 IF C4<7 THEN GOTO 2390
2380 LET C4=0
2382 GOTO 2330
2390 IF D(C4,4)=0 THEN GOTO 2360
2400 LET C(4)=D(C4,4)
2410 RETURN
3000 LET C(1)=D(C1,1)
3005 GOTO 2280
3020 GOSUB 9000
3025 PRINT AT 10,0;"ERRORE NELLA
 TABELLA DELLE POSSIBILITA"
3027 STOP
3030 LET C(2)=D(C2,2)
3035 GOTO 2320
3040 LET C(3)=D(C3,3)
3045 GOTO 2360
3100 GOSUB 9000
3102 PRINT AT 10,0;"HO INDOVINAT
O"
3104 PRINT
3106 PRINT "IN ";GU;" TENTATIVI"
3110 PAUSE 20000
3120 RETURN
3200 LET D(I,1)=0
3205 LET D(I,2)=0
3210 LET D(I,3)=0
3215 LET D(I,4)=0
3220 RETURN
4000 GOSUB 9000
4005 LET XI=XI+GU
4006 LET XT=XT+NO
4010 PRINT AT 7,0;"PARTITE"
4015 PRINT "███████"
4020 PRINT MN
4030 PRINT AT 12,0;"TOTALE TENTA
TIVI"
4035 PRINT "████████████████"
4042 PRINT "ZX81 ";XI
4043 PRINT N$;" ";XT
4050 PRINT AT 16,0;"COEFFICIENTE
 ABILITA"
4055 PRINT "████████████████████
█"
4062 PRINT "ZX81 ";MN/XI
4064 PRINT N$;" ";MN/XT
4070 PAUSE 20000
4080 RETURN
9000 CLS
9005 PRINT "████████████████████
████████████"
9010 PRINT
9020 PRINT TAB 10;"MASTER MIND"
9025 PRINT
9030 PRINT "████████████████████
████████████"
9040 RETURN
9500 DIM D(6,6)
9505 FOR K=1 TO 6
9510 FOR W=1 TO 4
9515 LET D(K,W)=K
9520 NEXT W
9530 NEXT K
9535 DIM T(4)
9537 DIM C(4)
9540 DIM E(6)
9542 DIM W(4)
9600 DIM G(20,6)
9605 RETURN
```

nella routine 9500-9580 servono proprio ad ovviare a questo inconveniente. Se nonostante tutto non riuscite a regolare il tocco e l'inconveniente persiste, dovete rinunciare al volo degli uccelli inserendo alle linee

Listato 8. *Questo listato rappresenta la conversione del programma MASTER MIND apparso sul numero 1 di* **Personal Software**. *A turno, il giocatore o il computer devono indovinare il numero pensato dall'avversario. Per ogni tentativo vanno indicati i NERI (numeri giusti al posto giusto) ed i BIANCHI (numeri giusti ma fuori posto).*
Il carattere CHR$ 3 (linea 635) si ottiene con Graphics e 7.

Il display file dello ZX81

Lo ZX81 conserva sempre in memoria una mappa aggiornata dello schermo.
Se date l'istruzione:

PRINT AT 10, 10; "A"

il computer modifica questa mappa facendo in modo che alle coordinate 10, 10 corrisponda il carattere A.

Ovviamente lo ZX81 non vede lo schermo come lo vedete voi: per lui non è costituito da un reticolo di 24 × 33 punti (32 colonne più il newline) bensì da una serie di 792 punti messi in fila più un punto finale, cioè 793 punti.

Il contenuto di ogni punto viene immagazzinato in un indirizzo di memoria: l'insieme dei 793 indirizzi costituisce il display file.

Quindi se ogni punto dello schermo è associato ad un indirizzo di memoria, leggendo l'indirizzo relativo al punto 10, 10 dovreste ottenere la lettera A. In realtà ottenete il numero 38, che è il numero di codice della lettera A.

Come fate a sapere quale sia l'indirizzo che contiene l'informazione relativa al punto 10, 10 dello schermo?

Anzitutto occorre stabilire il punto di partenza del display file: ammettiamo che il suo primo indirizzo sia il 17000.

In questo caso avrete che:

- all'indirizzo 17000 potete leggere il codice del carattere posto in 0, 0; (se non c'è niente leggete 0, cioè spazio);
- all'indirizzo 17001 potete leggere il codice del carattere posto in 0, 1;
- all'indirizzo 17000 + 34 potete leggere il codice del carattere posto in 1, 0;
- all'indirizzo 17000 + 341 potete leggere il contenuto del carattere posto in 10, 10.

Dovete aggiungere 341 perché per arrivare al carattere 10 nella linea 10 dovete attraversare:

- 10 linee intere (330 punti) + 11 punti

(linee e colonne si contano partendo da zero).

I problemi non sono ancora finiti. Infatti il display file non comincia ad un indirizzo fisso. L'indirizzo di inizio del display file è contenuto in altri due indirizzi. Con l'istruzione:

PEEK 16396 + 256 * PEEK 16397

otteniamo un numero equivalente al primo indirizzo del display file.

Con l'istruzione:

PEEK (PEEK 16396 + 256 * PEEK 16397)

otteniamo il numero di codice contenuto nel primo indirizzo del display file.

Con l'istruzione:

PEEK ((PEEK 16396 + 256 * PEEK 16397) + 341)

otteniamo il numero di codice del carattere posto in 10, 10.

Il discorso fatto con l'istruzione PEEK vale anche per l'istruzione POKE. Se vogliamo scrivere "A" nella posizione 10, 10 e vogliamo farlo in maniera assai più veloce di quella ottenibile con PRINT AT, possiamo scrivere:

```
LET B = PEEK 16396 + 256 * PEEK 16397
...
POKE B + 341, 38
```

L'istruzione POKE per la sua velocità di esecuzione viene spesso usata per modificare il display file nei giochi di movimento.

520 e 620 l'istruzione:

```
PAUSE 20000
```

Nel listato 9 viene fatto largo uso dell'operatore AND; per esempio nelle linee 610, 705, 760 e altre. Troverete una nota sugli operatori logici nella rubrica *I segreti dei personal*.

Versioni da 1 Kbyte

Infine non potevamo dimenticare gli amici sinclaristi con sistema in configurazione minima. Nei listati 10 e 11 troveranno la versione 1 Kbyte di un programma *Master Mind* e di *Almeno una volta*.

Nel listato 10 usiamo l'istruzione POKE 16418, 18 (vedi **Personal Software** n. 6: le variabili di sistema nello ZX81) che riserva le 18 linee nel basso dello schermo al computer, per cui possiamo utilizzare per la visualizzazione dell'output solo le sei linee più in alto. Se così non fosse, il computer darebbe errore di tipo 4 (manca spazio in memoria) quando i

```
 20 LET A$=""
 30 FOR K=PI/PI TO VAL "4"
 40 LET A$=A$+STR$ (INT (RND*VA
L "6")+PI/PI)
 50 NEXT K
 60 LET J=PI/PI
 70 PRINT "MASTER MIND "
 72 PRINT
 80 PRINT "TENT.";TAB VAL "6";"
NUM.";TAB VAL "11";"GIU";TAB VAL
 "16";"FP"
 85 PRINT J;TAB VAL "6";
 86 LET G=PI-PI
 87 LET F=PI-PI
 90 INPUT B$
 92 POKE VAL "16418",VAL "18"
100 IF LEN B$<>VAL "4" THEN GOT
O VAL "90"
105 PRINT B$;
106 IF B$=A$ THEN GOTO VAL "250
"
110 LET D$=A$
120 FOR K=PI/PI TO VAL "4"
125 IF D$(K)=B$(K) THEN GOSUB V
AL "300"
127 NEXT K
128 FOR K=PI/PI TO VAL "4"
130 FOR W=PI/PI TO VAL "4"
135 IF B$(K)=D$(W) THEN GOTO VA
L "350"
150 NEXT W
160 NEXT K
200 PRINT TAB VAL "11";G;TAB VA
L "16";F
210 LET J=J+PI/PI
220 IF J>=VAL "4" THEN SCROLL
230 GOTO VAL "85"
250 PRINT "[illegible]"
255 STOP
300 LET D$(K)="A"
302 LET B$(K)="B"
305 LET G=G+PI/PI
310 RETURN
350 LET D$(W)="A"
352 LET F=F+PI/PI
355 GOTO VAL "160"
```

Listato 10. *Il listato presenta una versione del gioco MASTER MIND che può girare su ZX81 con 1 Kbyte di memoria. PI intende pigreco (Function e M). L'uso delle funzioni VAL e PI consente un notevole risparmio di memoria. Se provate a correggere un errore in una linea quando il programma è caricato, noterete che il tasto EDIT non riesce a farla scendere in basso. Allora premete CLEAR e NEWLINE e riprovando EDIT vedrete che funziona.*

```
 10 LET A = VAL "26"
 15 CLS
 20 PRINT "ALMENO UNA VOLTA"
 30 PRINT "1 spazio nero e 1 bianco"; A; "1 spazio bianco e uno nero"
 42 PRINT
 50 PRINT "1METTI 2LEVI"
 60 PAUSE VAL "6E4"
 70 LET B$ = INKEY$
 72 IF B$ = "1" AND A < VAL "11" OR B$ = "1" AND A > VAL "30"
    THEN GOTO VAL "60"
100 PRINT AT VAL "4", PI-PI; "QUANTI ?"
110 PAUSE VAL "6E4"
120 LET C$ = INKEY$
122 IF B$ = "1" AND VAL C$ + A > VAL "31"
    THEN GOTO VAL "110"
124 LET J = PI/PI
130 GOSUB VAL "700"
140 PRINT AT VAL "2", PI-PI; "1 spazio nero e 1 spazio bianco";
    A: "1 spazio bianco e 1 spazio nero"
150 PRINT AT VAL "4", PI-PI; "PREMI PER ME"
160 PAUSE VAL "4E4"
170 FOR K = VAL "1" TO VAL "31" STEP VAL "5"
180 FOR W = K TO K + VAL "4"
190 IF W = A THEN GOTO VAL "220"
200 NEXT W
210 NEXT K
220 LET B$ = "2"
230 LET C$ = STR$ (A-K)
235 LET J = VAL "2"
240 GOSUB VAL "700"
250 GOTO VAL "15"
700 LET A = A + (VAL C$ AND B$ = "1") - (VAL C$
    AND B$ = "2")
702 IF A = PI/PI THEN PRINT AT VAL "2", VAL "2"; "1";
    AT VAL "4", PI-PI; ("HAI VINTO TU" AND J =
    VAL "1") + ("HO VINTO IO" AND J = VAL "2")
703 IF A = PI/PI THEN STOP
705 RETURN
```

Listato 11. *Versione 1 Kbyte del programma ALMENO UNA VOLTA. Caricando il programma battete per prima la linea 702. I caratteri sottolineati vanno scritti in inverso. Le stringhe: 1METTI 2LEVI (50), QUANTI ? (100), HAI VINTO TU, HO VINTO IO contengono tutte 12 caratteri in modo che sovrapponendosi si cancellano.*

tentativi fossero molti e venissero utilizzate più di 6 linee.

La linea 220 esegue lo scroll, che naturalmente ha effetto dalla linea 5 in su. La colonna GIÙ indica i numeri giusti al posto giusto, la colonna FP i numeri giusti fuori posto.

Il listato 11 conserva la quantità di cubetti nella variabile A, di cui viene stampato il valore. Il giocatore può ancora togliere o aggiungere da 1 a 4 cubetti, il computer toglie solamente. Chi toglie l'ultimo perde.

Nella rubrica *I segreti dei personal* troverete informazioni sulle tecniche di risparmio di memoria usate in questi due programmi.

Caricandoli non dovete aggiungere né cambiare assolutamente niente, per non provocare errori di tipo 4, dal momento che la memoria viene riempita al suo limite. ■

I programmi di simulazione

Un'introduzione ad un utilizzo creativo del vostro personal

di *Francesco Sardo*

La maggior parte dei fenomeni fisici sono descrivibili mediante espressioni che mettono in relazione quantitativamente le grandezze in gioco, dando la possibilità di calcolare le variazioni di una al variare delle altre. Se il fenomeno fisico è quindi esattamente descrivibile, è anche simulabile mediante le stesse espressioni.

Se, per esempio, si assume che un solido cada nel vuoto da 100 m di altezza, mediante la legge che descrive la caduta dei gravi è possibile sapere ad ogni tempo che velocità ha raggiunto il corpo, e a che distanza si trova dal suolo.

Questa caduta può essere quindi simulata al calcolatore; dando solo l'altezza da cui inizia la caduta, il calcolatore potrà realizzare una tabella tempi/velocità/spazio percorso, e alla fine, dare l'energia del suo urto col suolo.

È chiaro come ciò rappresenti, con opportuni accorgimenti grafici, un'ottima illustrazione del fenomeno, che permetterà, di esaminarlo nei vari dettagli.

Ogni programma di simulazione è costituito dalle seguenti parti:

1. Messaggi iniziali, con enunciazione delle regole, descrizione del fenomeno, modalità d'uso del programma.
2. Visualizzazione della situazione di partenza.
3. Input dei dati (nel ns. esempio, solo altezza iniziale).
4. Calcolo delle modificazioni della situazione avvenuta, effettuato mediante le equazioni che descrivono il fenomeno (es.: nuova altezza o nuova velocità).
5. Visualizzazione della nuova situazione.
6. Verifica dell'eventuale raggiungimento delle condizioni finali:
   - in caso affermativo, messaggi di chiusura,
   - in caso negativo, ritorno al punto 3 o al punto 4 nel caso in cui si devono immettere nuovi dati o meno.

Esaminiamo ora le principali applicazioni di questo tipo di programmi.

Giochi

Esistono o sono immaginabili un gran numero di giochi di simulazione: la guida di un aereo, la caccia ad un sottomarino con una nave di superficie, le manovre del modulo lunare o dello Shuttle.

Anche l'andamento della Borsa valori può essere simulato: anche se il rialzo e il ribasso dei titoli non dipendono da alcuna legge matematica, l'andamento generale può essere espresso con buona approssimazione mediante funzioni di tipo sinusoidale.

La buona qualità del gioco dipende poi dalla veste grafica, dalla fantasia del programmatore, dalla accu-

ratezza con cui si tiene conto dei vari fattori che prendono parte al fenomeno.

Ad esempio, l'atterraggio di un aereo può essere simulato in maniera approssimativa dando come variabili solo l'accelleratore e la posizione degli alettoni, o in maniera accurata, tenendo conto del peso dell'aereo, delle riserve di carburante, della velocità di stallo, del vento, ecc.

È chiaro che l'efficacia della visualizzazione è determinante.

Agli effetti della piacevolezza del gioco, è più importante far vedere gli strumenti di bordo e la pista, e dare l'impressione di partecipare attivamente all'operazione, piuttosto che tener conto di dieci o quindici equazioni che collegano la velocità con la posizione dell'accelleratore o la quota con la posizione degli alettoni o la riserva di carburante con il regime dei motori e il tempo di volo o l'assetto dell'aereo con altre variabili ancora.

La maggior parte delle equazioni usate saranno quindi empiriche o approssimate.

Insegnamento

I programmi di simulazione, come dicevamo a proposito della legge sulla caduta dei gravi, hanno una efficacia ineguagliabile nell'illustrare didatticamente un fenomeno.

Le applicazioni sono le più svariate, praticamente ogni fenomeno fisico può essere visualizzato e illustrato mediante un programma di simulazione.

Esistono già programmi che illustrano le leggi dell'ottica, la rifrazione, la interferenza, i fenomeni oscillatori, la legge di Ohm. Un programma simula alla perfezione il rimbalzare di una palla di gomma, calcolando punto per punto la sua velocità e la sua altezza dal suolo, e visualizzandola sullo schermo o sulla stampante. Un buon programma didattico naturalmente deve non solo far vedere la palla che rimbalza, perché ciò si può più efficacemente vedere con una palla reale, ma mostrare le equazioni usate, i calcoli effettuati e i risultati ad ogni istante, realizzando così una traccia che è poi il cammino della palla costruito punto per punto.

Si può far vedere il comportamento di un gas, e come esso risponda alle variazioni di temperatura, di pressione o di volume indotte dall'utente, a seconda che sia un gas ideale o un gas reale.

Si possono simulare modificazioni a temperatura costante, o a pressione costante o a volume costante, esaminando quindi e illustrando tutte le leggi dei gas.

Nel campo della chimica, fra le altre cose, si può illustrare la cinetica delle reazioni osservando lo scomparire dei reagenti e il formarsi dei prodotti al passare del tempo, e in funzione delle concentrazioni dei composti in gioco, a seconda dell'ordine della reazione.

A tale scopo è opportuno dividere lo schermo in più aree: in una si può visualizzare il progressivo scomparire dei reagenti e apparire dei prodotti; in un'altra i dati calcolati ad ogni istante, in una terza la reazione e l'espressione con cui vengono calcolati i dati.

Nel campo della matematica si può visualizzare la distribuzione statistica di eventi casuali simulati mediante la funzione RND, come ad esempio il lancio di dadi.

Nel campo della ecologia, l'aumento o la diminuzione relativa della popolazione di due specie in competizione (classico, le volpi e i conigli, vedi cassetta dimostrativa dello Spectrum).

E così proseguendo si potrebbero fare centinaia di esempi; starà ai programmatori sbizzarrirsi in base alla loro fantasia e alle esigenze dei discenti.

Programmi applicativi

Il classico programma applicativo di questo tipo è il simulatore di volo; poiché un errore nella guida di un aereo può essere estremamente pericoloso, gli allievi piloti si allenano sui simulatori. Questi programmi sono molto accurati, e tengono conto di tutti i fattori che possono influenzare il volo, inserendo anche delle situazioni di pericolo.

Ma a parte questo uso un po' particolare, i programmi di simulazione vengono utilizzati tutti i giorni; anche i programmi di calcolo di strutture, visto che lavorano mediante verifica, possono essere considerati di simulazione; si ipotizza una struttura, la si simula mediante le equazioni che la descrivono, si vede se regge. Se così non avviene, senza scomodare squadre di soccorso e ambulanze, si prova una struttura appena più resistente.

L'applicazione è comunque generale e di estremo interesse: un impianto petrolchimico, un acquedotto, un impianto di depurazione possono essere descritti mediante una serie di equazioni che collegano fra loro tutte le variabili in gioco; in questo caso si parla di *modello matematico* del sistema.

Un modello può essere più o meno accurato, a seconda se si tiene conto di tutte le variabili o se se ne trascura qualcuna. Al variare di qualche grandezza, si vede come risponde tutto il sistema. Si calcolano le condizioni di costo minimo e di funzionamento ottimale.

È bene notare come anche in questi casi, sebbene il programma possa diventare anche molto complesso per quel che riguarda il calcolo, la sua struttura rimanga sempre quella descritta inizialmente.

Bisogna però tener presente che è necessario esplicitare nel programma la funzione da calcolare; cioè l'espressione per determinare y deve essere nella forma $y = f(x)$, e non può essere nella forma $f(x, y) = 0$.

Occorre quindi che sia stabilito già all'atto della preparazione del programma, quali debbano essere le variabili indipendenti e quali le variabili dipendenti.

Allora ne risulta che è necessario impostare il programma di simulazione mediante un modello matematico in due modi diversi, a seconda dello scopo a cui deve servire:

a) Se l'obiettivo è il dimensionamento di un nuovo impianto, occorre dare come input le prestazioni richieste alle condizioni di funzionamento prescelte ed esplicitare nel programma, le variabili di dimensionamento (volu-

mi, spessori, diametri, potenze), per poterle avere in output.

b) Se l'obiettivo è il controllo di un impianto esistente, e l'esame della sua risposta al mutare delle condizioni operative, occorre dare come input le dimensioni, e in generale tutte le grandezze che risultano già definite per il fatto stesso che il sistema esiste e specificare quelle che sono soggette a regolazioni, che potranno essere esplicitate durante il funzionamento (portate, concentrazioni, rese).

Sarà così possibile, dato un sistema definito ed esistente, studiarne il comportamento nelle sue variabili di funzionamento al variare delle regolazioni fino ad identificare le condizioni ottimali di massima resa.

Non è, in generale, possibile impostare un programma in modo che serva, senza modifiche, sia per il dimensionamento in fase di progettazione che per il controllo del funzionamento, a meno che esso non consista in due programmi concatenati da usare in alternativa. ■

Il tanto atteso Microdrive Sinclair è arrivato!

Presentato alla stampa di settore il nuovo ZX Microdrive, che adottando un rivoluzionario tipo di supporto magnetico e avanzate caratteristiche di progetto, costituisce una novità per la soluzione del delicato problema della memorizzazione programmi e dati nel settore degli home computer.
Lo ZX Microdrive consente infatti di registrare, su di una piccola cartuccia magnetica intercambiabile, ben 85 Kbyte di informazioni.
Ogni cartuccia può contenere un massimo di 50 file che sono identificati da un nome convenzionale scelto dall'utente. I file possono essere singolarmente caricati, memorizzati, visualizzati in ordine alfabetico o cancellati. Il tempo medio di accesso ad un file è dichiarato essere di circa 3,5 secondi, un notevole miglioramento rispetto ad una normale cassetta audio usata dalla maggior parte degli home computer in commercio. Il circuito di controllo del Microdrive, contenuto nella ZX Interfaccia 1 espande il linguaggio Sinclair BASIC residente nel microcomputer ZX Spectrum, includendo la gestione dei file e comandi per il colloquio fra microelaboratori.
Ciò rende il BASIC valido non più solo come linguaggio di programmazione ma anche come sistema operativo. Fino ad otto Microdrive possono essere collegati ad uno ZX Spectrum tramite la ZX Interfaccia 1, dando un totale di 680 Kbyte di informazioni sempre in linea.
Lo ZX Microdrive amplia le possibilità dello ZX Spectrum in quei settori, come quello della didattica e della piccola gestione di dati, dove è necessaria una veloce ricerca delle informazioni memorizzate su un supporto magnetico.
Le caratteristiche tecniche fondamentali sono le seguenti:

- Comandi di SAVE, LOAD e VERIFY per la memorizzazione, il caricamento e la verifica dei programmi.
- Comando FORMAT per l'inizializzazione delle cartucce magnetiche.
- Comando CAT per ottenere su video la lista in ordine alfabetico di tutti i file contenuti nella cartuccia. Viene fornita anche una indicazione dello spazio libero disponibile su cartuccia.
- Alimentazione attraverso lo Spectrum.
- Spia luminosa rossa che indica lo stato di funzionamento.

La ZX Interfaccia 1, che incorpora anche una interfaccia RS 232 e un sistema di collegamento in rete locale, si connette alla parte posteriore dello ZX Spectrum, permettendo comunque il collegamento di altre espansioni periferiche dello ZX Spectrum. La interfaccia seriale RS 232, standard industriale universalmente adottato, permette il collegamento fra lo ZX Spectrum ed una ampia gamma di periferiche e di altri computer dotati della medesima interfaccia. Grazie alla RS 232 è anche possibile trasmettere dati su linea telefonica utilizzando un modem.
L'interfaccia opera con velocità selezionabile via software con tutti i valori standard di trasmissione fino al massimo di 19.200 baud.
Il collegamento in rete locale è costituito da una linea di comunicazione ad alta velocità che può collegare fino a 64 Spectrum, trasmettendo a 100 Kbaud.
Tutte le immagini contenute in uno schermo video possono essere trasferite in circa 3 secondi e il protocollo di collegamento permette ad ogni stazione della rete di specificare quali sono le stazioni trasmittenti e riceventi.
È inoltre possibile diffondere un messaggio ad ogni ZX Spectrum collegato alla rete realizzando un interessante sistema di broadcasting.
Ogni Sinclair ZX Spectrum può agire come servitore di altri ZX Spectrum dalla rete pilotando una stampante ZX o qualsiasi altra periferica collegata tramite la interfaccia RS 232.
Ogni ZX Spectrum può inviare e ricevere file dagli altri computer della rete sfruttando al massimo le possibilità offerte dallo ZX Microdrive.
La rete locale ZX offre un numero molto vasto di possibilità agli utilizzatori realizzando a costi molto contenuti un sistema multiutente di scambio di informazioni. Le applicazioni possibili possono essere individuate sia nel settore della didattica e dell'ufficio sia in quello dell'intrattenimento dove sono realizzabili sfide multiutente.



Il nuovo ZX Microdrive: compatto e ad un prezzo estremamente contenuto offre a tutti gli utenti dello ZX Spectrum i vantaggi di una memoria di massa veloce ed affidabile.



Il Sinclair ZX Microdrive collegato allo ZX Spectrum grazie alla ZX Interfaccia 1.

SFOGLIA JACKSON

GRUPPO
EDITORIALE
JACKSON

RIVISTE
ENCICLOPEDIE
LIBRI
MANUALI

Trovi il meglio dell'elettronica e dell'informatica



**GRUPPO
EDITORIALE
JACKSON**

Campagna Abbonamenti 1984 - Spedizione in abbonamento postale gruppo III/70

ABBONARSI CONVIENE

Uno sconto sicuro, per chi sottoscrive un abbonamento

Riviste	Uscite	Importo Globale	Costo abbonamento	Costo abbonamento estero
Personal Software	**10**	~~L. 35.000~~	**L. 28.000**	**L. 42.000**
Bit	**11**	~~L. 44.000~~	**L. 35.000**	**L. 52.500**
Informatica Oggi	**11**	~~L. 33.000~~	**L. 27.000**	**L. 40.500**
Elektor	**12**	~~L. 36.000~~	**L. 29.000**	**L. 43.500**
Automazione Oggi	**11**	~~L. 33.000~~	**L. 26.000**	**L. 39.000**
Elettronica Oggi	**11**	~~L. 38.500~~	**L. 31.000**	**L. 46.500**
L'Elettronica	**22**	~~L. 55.000~~	**L. 44.000**	**L. 66.000**
Telecomunicazioni	**8**	~~L. 28.000~~	**L. 22.000**	**L. 33.000**
Video Giochi	**11**	~~L. 33.000~~	**L. 25.000**	**L. 37.500**
Strumenti musicali	**10**	~~L. 30.000~~	**L. 24.000**	**L. 36.000**

Un supersconto a chi si abbona a due o più riviste

Tutti coloro che sottoscrivono l'abbonamento a due o più riviste godono di un prezzo ulteriormente agevolato, come appare nella seguente tabellina.

Esempio: Bit + Informatica Oggi
L. 35.000 + L. 27.000 = L. 62.000 **meno** L. 2.000 = L. 60.000!

Abbonamento a 2 riviste:	L. 2.000 ***in meno*** sulla somma dei 2 prezzi d'abbonamento
Abbonamento a 3 riviste:	L. 4.000 ***in meno*** sulla somma dei 3 prezzi d'abbonamento
Abbonamento a 4 riviste:	L. 7.000 ***in meno*** sulla somma dei 4 prezzi d'abbonamento
Abbonamento a 5 riviste:	L. 10.000 ***in meno*** sulla somma dei 5 prezzi d'abbonamento
Abbonamento a 6 riviste:	L. 13.000 ***in meno*** sulla somma dei 6 prezzi d'abbonamento
Abbonamento a 7 riviste:	L. 16.000 ***in meno*** sulla somma dei 7 prezzi d'abbonamento
Abbonamento a 8 riviste:	L. 20.000 ***in meno*** sulla somma degli 8 prezzi d'abbonamento
Abbonamento a 9 riviste:	L. 25.000 ***in meno*** sulla somma dei 9 prezzi d'abbonamento
Abbonamento a 10 riviste:	L. 30.000 ***in meno*** sulla somma dei 10 prezzi d'abbonamento

Un premio a sorte per ogni rivista

Ad ogni Rivista JACKSON sono abbinati uno o più premi prestigiosi e di grande valore, da estrarre a sorte tra gli abbonati della relativa testata. Eccone l'elenco:

Personal Software	**3 Personal Computer Sinclair Spectrum distribuiti da G.B.C.-Rebit**
Bit	**1 Personal Computer IBM**
Informatica Oggi	**1 Personal Computer IBM**
Elektor	**1 oscilloscopio UNAOHM doppia traccia, mod. 4001 B DT**
Automazione Oggi	**1 Personal Plotter M 84 della Calcomp**
Elettronica Oggi	**1 Oscilloscopio PM3215 Philips**
L'Elettronica	**2 Personal Computer portatili Epson HX-20, distribuiti da SEGI**
Telecomunicazioni	**3 Telefoni Margherita e 3 segreterie telefoniche della ITALTEL**
Video Giochi	**5 basi ATARI**
Strumenti Musicali	**1 Chitarra elettr. B.C. Rich mod. New Jersey Ser. Eagle, distr. da Meazzi (MI)**

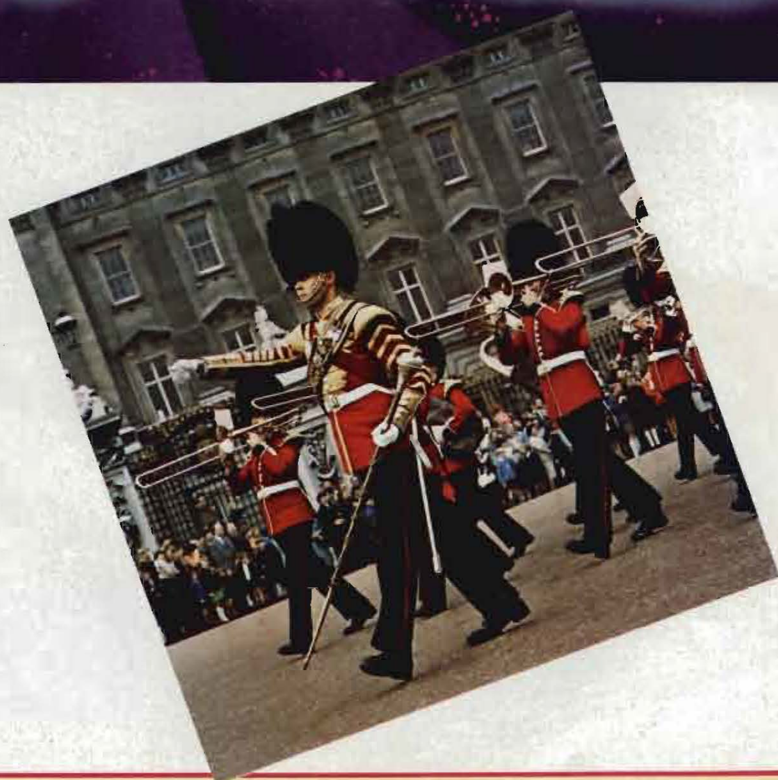
Abbonandosi a più Riviste, inoltre, si ha diritto a partecipare a più estrazioni e a far inserire il proprio nominativo un equivalente numero di volte tra i partecipanti all'estrazione del **Superpremio JACKSON "3 viaggi a Londra"**

Per abbonarsi
Per sottoscrivere abbonamenti utilizzate il modulo di c.c.p. n. 11666203 intestato a Jackson - Milano inserito in questo fascicolo, oppure inviate un assegno o un vaglia postale al nostro ufficio abbonamenti.

leggi Jackson e Vinci

CAMPAGNA ABBONAMENTI 1984

Anche quest'anno la Campagna Abbonamenti JACKSON riserva a tutti i lettori abbonati — oltre a tariffe preferenziali — anche un grande CONCORSO con moltissimi premi.
Ad ogni Rivista JACKSON, ad esempio, sono abbinate una o più apparecchiature di elettronica o di informatica da estrarsi a sorte tra tutti gli Abbonati di quella testata.
Abbonarsi a più Riviste significa, quindi, partecipare di diritto a più estrazioni e perciò aumentare le proprie possibilità di vittoria.
Condizione essenziale per partecipare alle estrazioni in programma è inviare il proprio abbonamento entro il 28-2-1984.
Tutti gli Abbonati alle Riviste JACKSON, infine, parteciperanno all'estrazione generale che vedrà in palio il Superpremio JACKSON, consistente in 3 viaggi a Londra, per 2 persone, della durata di 5 giorni.



REGOLAMENTO DEL CONCORSO

1) Il Gruppo Editoriale JACKSON S.r.l. in occasione della Campagna Abbonamenti 1984, promuove un grande Concorso a premi.
2) Per partecipare è sufficiente sottoscrivere un abbonamento a una qualsiasi delle 10 Riviste JACKSON entro il 28-2-'84.
3) Fra gli Abbonati di ogni Rivista saranno sorteggiati uno o più premi specifici, come indicato nelle singole condizioni di abbonamento. Tra tutti gli Abbonati del Gruppo JACKSON, infine, saranno sorteggiati 3 viaggi a Londra per 2 persone, con soggiorno di 5 giorni.
4) Gli Abbonati a più di una Rivista JACKSON, oltre a partecipare alle estrazioni dei premi abbinati alle testate, avranno diritto all'inserimento del proprio nominativo, per l'estrazione relativa al viaggio-soggiorno, tante volte quante sono le Riviste a cui sono abbonati.
5) L'estrazione dei premi indicati avverrà presso la Sede JACKSON entro il 30-6-'84.
6) L'elenco dei vincitori e dei relativi premi sarà pubblicato su almeno 6 Riviste JACKSON. Il Gruppo Editoriale JACKSON, inoltre, ne darà comunicazione scritta ai singoli vincitori.
7) I premi verranno messi a disposizione degli aventi diritto entro 60 giorni dalla data dell'estrazione.
8) I dipendenti, i familiari, i collaboratori del Gruppo Editoriale JACKSON sono esclusi dal presente Concorso.

Aut. Min. D.M. N° 4/247403 del 27-10-83

BIT

La più letta delle Riviste Jackson, la prima Rivista europea di personal computer, software e accessori. Ogni numero rappresenta un'affascinante avventura nel mondo dei piccoli sistemi, un universo in cui si muove e opera un numero sempre crescente di hobbisti, tecnici, professionisti, appassionati, ai quali BIT dedica idee e soluzioni pratiche per imparare a programmare, per 'giocare' con il computer, per lavorarci

In ogni numero

Bitest
Novità del mercato
Software
Articoli tecnici
Analisi e raffronti
64 pagine di programmi

Riviste complementari

Personal Software
Informatica Oggi

Abbonarsi conviene

11 numeri all'anno di cui 4 monografici
***solo* L. 35.000**
invece di L. 44.000



In più, gli abbonati di BIT partecipano all'estrazione di un favoloso Personal Computer della IBM

... e a quella del Superpremio JACKSON "3 viaggi a Londra"

PERSONAL SOFTWARE


Mini Flexible Disc 5 1/4" certificato Error Free per impiego su drive a due testine, a doppia densità (2D), 40 tracce, con anello di rinforzo.

1000 dischetti per 1000 Abbonati

I primi 1000 Abbonati di Personal Software riceveranno, insieme al primo numero della Rivista, un Mini Flexible Disc della Memorex. Abbonarsi conviene. Subito è meglio!

Abbonarsi conviene

10 numeri all'anno
***solo* L. 28.000**
invece di L. 35.000

Personal Software

La Rivista che ha inaugurato un capitolo nuovo nell'editoria europea dedicata al software per personal computer. Attualissima, interattiva col lettore (sia quello che già possiede un personal, sia quello che intende acquistarlo) Personal Software deve la sua fortuna a una formula espositiva del tutto originale, che fornisce agli 'amatori' — accanto ad articoli generali e di approfondimento teorico — anche tutta una serie di programmi già predisposti e testati

In ogni numero
Aspetti e problemi del software
Programmi e sistemi operativi
Speciale

Riviste complementari
Bit



In più, gli abbonati di Personal Software partecipano all'estrazione di 3 Personal Computer Sinclair Spectrum distribuiti da G.B.C. - Rebit

... e a quella del Superpremio JACKSON "3 viaggi a Londra"

Video Giochi

Nata nel gennaio 1983, questa Rivista è subito diventata, per migliaia di appassionati, la guida indiscussa al fantastico mondo dei videogames. Ricchissima di notizie, indicazioni, spiegazioni, suggerimenti... aperta al dialogo e alle 'provocazioni' di lettori giovanissimi e non, Video Giochi rappresenta una vera sfida alla fantasia, alla capacità di cimentarsi in entusiasmanti duelli elettronici, in una gara ideale con avversari di tutto il mondo. Video Giochi, infatti, dal punto di vista delle anticipazioni di mercato è una Rivista veramente internazionale, il 'pulpito' da cui tutte le più grandi Industrie mondiali di videogames parlano a milioni di fans. Purtuttavia, Video Giochi non è solo svago fine a se stesso, ma anche la dimostrazione di come, attraverso il divertimento, si possa arrivare ad essere 'esperti programmatori'. Perché il mondo dei computer ha tante facce e quella di Video Giochi, pur essendo la più eccitante, non è per questo la meno istruttiva.

In ogni numero

Prezzi e novità
Schemi di programmazione
Il mercato
Video-Gare

Riviste complementari

Personal Software-Bit

Abbonarsi conviene

11 numeri all'anno
***solo* L. 25.000**
invece di L. 33.000



In più, gli abbonati di Video Giochi partecipano all'estrazione di 5 Basi ATARI

... e a quella del Superpremio JACKSON "3 viaggi a Londra"



BIBLIOTECA JACKSON

Un settore diversificato e in continua evoluzione come quello dell'elettronica e dell'informatica non può prescindere dal libro specializzato. Il Gruppo Editoriale JACKSON, anche in questo campo, rappresenta il punto di riferimento più qualificato per quanti non si accontentino di un'informazione effimera e desiderino approfondire i mille argomenti legati al mondo del computer e della microelettronica.
Oltre 140 sono i volumi che costituiscono attualmente la 'Biblioteca JACKSON' e, annualmente, ne vengono stampate e vendute oltre 150 mila copie. Sono opere di specialisti, contributi di grandi Centri di Ricerca, manuali per neofiti o testi di fondamentale importanza per tecnici e operatori.
In queste pagine, al fine di facilitare la ricerca, i titoli della 'Biblioteca JACKSON' sono stati suddivisi in 10 grandi famiglie d'argomenti.
Aprono questa panoramica gli 8 tomi di E.I. la grande Enciclopedia di Elettronica e di Informatica, l'opera che ha rappresentato il 'caso' editoriale dell'anno. Prima e unica al mondo, E.I. è stata realizzata dalla JACKSON in collaborazione con il Learning Center Texas Instruments. Pubblicata a fascicoli settimanali, E.I. ha suscitato l'entusiasmo di decine di migliaia di lettori. Evidentemente di un'opera così, seria e completa, si avvertiva davvero il bisogno e è merito della JACKSON l'aver saputo interpretare e dar corpo ai desideri di un vastissimo pubblico.
Oggi tutti i volumi della 'Biblioteca' vengono proposti ai lettori con una formula d'acquisto che prevede consistenti agevolazioni per gli abbonati delle Riviste JACKSON. Ma, al di là dell'occasione contingente, resta il fatto che la 'Biblioteca JACKSON', nelle sue molteplici sfumature e specializzazioni, rappresenta un momento fondamentale di aggiornamento e approfondimento per quanti vivono nel mondo dell'elettronica e dell'informatica.
Un mondo di cui la JACKSON conosce tutti i segreti.

LEGGERE JACKSON
sempre necessario, oggi ancora più conveniente
Sconto 20% sui prezzi di copertina
a tutti gli Abbonati delle Riviste JACKSON

Informatica

TELEMATICA
Dal viewdata all'office automation, un panorama dei problemi teorico-pratici di questa nuovissima disciplina
286 pag. **L. 19.000**
Cod. 518D

MICROELETTRONICA
I come e i perché della nuova Rivoluzione industriale e lo scenario tecnico-economico-sociale del prossimo ventennio
180 pag. **L. 11.500**
Cod. 315P

COMPUTER GRAFICA
L'immagine come informazione e il calcolatore come produttore d'immagine
174 pag. **L. 29.000**
Cod. 519P

DIZIONARIO DI INFORMATICA
15000 termini inglese-italiano-tedesco;italiano-inglese; tedesco-inglese
920 pag. **L. 55.000**
Cod. 100H

VOI E L'INFORMATICA
In 100 tavole: gli strumenti dell'informatica,l'Informatica e l'Azienda; realtà e prospettive tecnologiche in modo sintetico, rigoroso ma completo
116 pag. **L. 15.000**
Cod. 526A

PRINCIPI E TECNICHE DI ELABORAZIONE DATI
Per l'autoapprendimento dei basilari principi di flusso e di gestione nei sistemi di elaborazione elettronica
254 pag. **L. 17.000**
Cod. 309A

ELEMENTI DI TRASMISSIONE DATI
Un valido ausilio per tecnici e studenti che vogliono approfondire le tecniche di comunicazione
178 pag. **L. 10.500**
Cod. 316D

MICROPROCESSORI AL SERVIZIO DEL MANAGEMENT
CAD/CAM e robotica la loro applicazione in Azienda, l'impatto su qualità e produttività,le prospettive
292 pag. **L. 20.000**
Cod. 335H

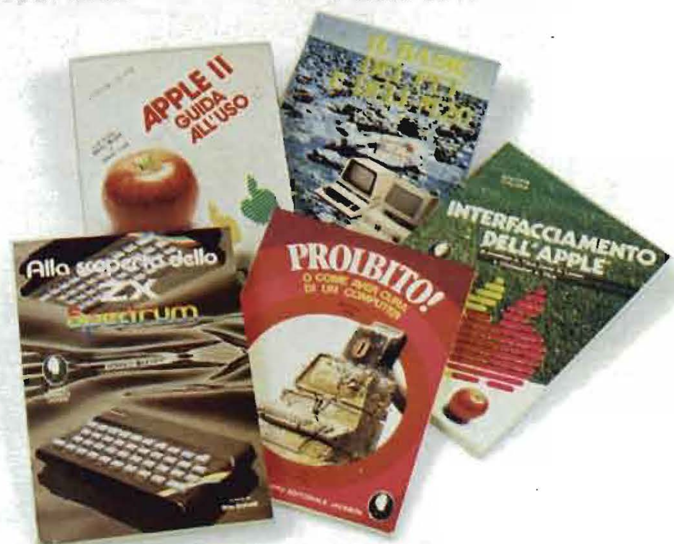


Personal e home computer

INTRODUZIONE AL PERSONAL E BUSINESS COMPUTING
Un approccio semplice ed esauriente al mondo del microcomputer: funzionamento, programmazione, scelta dei sistemi
224 pag. **L. 14.000**
Cod. 303D

GUIDA AL SINCLAIR ZX81-ZX80 E NUOVA ROM
Per avvicinarsi all'informatica e apprendere i segreti della programmazione in BASIC
262 pag. **L. 16.500**
Cod. 318B

APPLE II GUIDA ALL'USO
Per imparare a conoscere e usare uno dei sistemi più diffusi al mondo
400 pag. **L. 26.000**
Cod. 331P



INTRODUZIONE AI MICROCOMPUTER
Vol. 0: Il libro del principiante
240 pag. **L. 16.000**
Cod. 304A
Vol. I: Il libro dei concetti fondamentali
320 pag. **L. 18.000**
Cod. 305A

IL BASIC DEL PET E DELL'M20
Un validissimo supporto e strumento di lavoro per chiunque voglia o debba imparare a programmare in BASIC con un Commodore o un Olivetti M20
232 pag. **L. 16.000**
Cod. 336D

IMPARIAMO A PROGRAMMARE IN BASIC CON IL VIC/CBM
176 pag. **L. 12.500**
Cod. 507A

IMPARIAMO A PROGRAMMARE IN BASIC CON IL PET/CBM
L'informatica a disposizione di tutti, senza inutili teorizzazioni e tanta pratica
180 pag. **L. 11.500**
Cod. 506A

INTERFACCIAMENTO DELL'APPLE
Il libro indispensabile a un uso 'estremo' dell'APPLE: controllo di dispositivi, temperature, soglie luminose, liquidi...
208 pag. **L. 14.000**
Cod. 334B

ALLA SCOPERTA DELLO ZX SPECTRUM
Le grandi possibilità del più piccolo dei microcomputer Sinclair
320 pag. **L. 22.000**
Cod. 337B

PROIBITO! COME AVER CURA DI UN COMPUTER
Tutto quello che bisogna sapere per non mandare in tilt un calcolatore
208 pag. **L. 14.000**
Cod. 333D

APPLE - MEMO
Sintassi dei comandi, codici caratteri, messaggi di errore, linguaggio macchina e tante altre utili informazioni
150 pag. **L. 15.000**
Cod. 340H

Linguaggi di programmazione

PROGRAMMARE IN BASIC
Caratteristiche e peculiarità del BASIC applicato a: Apple, PET, TRS 80
94 pag. **L. 8.000**
Cod. 513A

IL BASIC E LA GESTIONE DEI FILE
Vol. I: Metodi pratici
Dal BASIC microsoft, ai metodi pratici, ai messaggi d'errore
164 pag. **L. 11.000**
Cod. 515H

COME PROGRAMMARE
Tutte le fasi di una corretta programmazione in BASIC o in qualsiasi altro linguaggio
192 pag. **L. 12.000**
Cod. 511A

INTRODUZIONE AL BASIC
Tecnica e pratica in un libro che costituisce un vero e completo corso di BASIC
314 pag. **L. 21.000**
Cod. 502A











IL BASIC PER TUTTI
Per i neofiti una facile e immediata introduzione al linguaggio BASIC e al mondo dei calcolatori
264 pag. **L. 17.500**
Cod. 525A

PROGRAMMARE IN PASCAL
Tutti i vantaggi di un linguaggio sempre più importante e diffuso
208 pag. **L. 14.000**
Cod. 514A

INTRODUZIONE AL PASCAL
Per conoscere, capire, usare il linguaggio destinato a spodestare il FORTRAN, l'ALGOL, il PL/I ecc.
484 pag. **L. 30.000**
Cod. 516A

IMPARIAMO IL PASCAL
Consigli, problemi, esercizi per l'autoapprendimento del PASCAL. Divulgazione senza pedanterie
162 pag. **L. 11.500**
Cod. 501A

PASCAL MANUALE E STANDARD DEL LINGUAGGIO
Dagli Autori del Pascal, il "libro" sul Pascal
186 pag. **L. 11.500**
Cod. 500P

DAL FORTRAN IV AL FORTRAN 77
Per chi deve programmare a livello tecnico scientifico e per chi vuole approfondire le conoscenze del linguaggio
266 pag. **L. 18.000**
Cod. 517P

CP/M con MP/M
Un libro destinato a rendere semplice l'uso del CP/M e, automaticamente, quello dei microcomputer
320 pag. **L. 22.000**
Cod. 510P

IL FORTH PER VIC 20 E CBM 64
La programmazione in FORTH e la sua implementazione sui Commodore VIC 20 e CBM 64
150 pag. **L. 11.000**
Cod. 527B

PROGRAMMARE IN ASSEMBLER
Il manuale pratico che aspettavano hobbisti e utenti di personal computer
160 pag. **L. 10.000**
Cod. 329A

GUIDA ALLA PROGRAMMAZIONE IN ASSEMBLER Z80 SUL PICO COMPUTER
Di esercizio in esercizio fino a diventare esperti programmatori
138 pag. **L. 9.000**
Cod. 330D



Programmi

66 PROGRAMMI PER ZX81 E ZX80 CON NUOVA ROM + HARDWARE
Come sfruttare tutte le capacità degli ZX e, addirittura, moltiplicarle
144 pag. **L. 12.000**
Cod. 520D

50 ESERCIZI IN BASIC
Una raccolta completa e progressiva di esercizi matematici, gestionali, operativi, statistici, di svago
208 pag. **L. 13.000**
Cod. 521A

GIOCARE IN BASIC
Il gioco come metodo d'apprendimento del BASIC e dei microcomputer
324 pag. **L. 20.000**
Cod. 522A

PROGRAMMI DI MATEMATICA E STATISTICA
Come acquistare la logica necessaria a risolvere con metodo, senza perdite di tempo, i problemi con il calcolatore
228 pag. **L. 16.000**
Cod. 552D

PROGRAMMI PRATICI IN BASIC
Programmi di tipo finanziario, matematico, scientifico, manageriale... già pronti e sperimentati
200 pag. **L. 12.500**
Cod. 550D

77 PROGRAMMI PER SPECTRUM
Dalla Grafica alla Business Grafica, dalla musica alle animazioni, dai giochi all'Elettronica...tutte le possibilità offerte dallo Spectrum
150 pag. **L. 16.000**
Cod. 555A

75 PROGRAMMI IN BASIC PER IL VOSTRO COMPUTER
Programmi sperimentati e pronti da usare, oppure da rielaborare, ampliare, modificare, assemblare
196 pag. **L. 12.000**
Cod. 551D

SOLUZIONI DI PROBLEMI IN PASCAL
Un approccio disciplinato alla soluzione di problemi col calcolatore e un modo garantito di imparare a programmare
450 pag. **L. 28.000**
Cod. 512P

PROGRAMMI SCIENTIFICI IN PASCAL
Per costruirsi una 'libreria' di programmi in grado di risolvere i più frequenti problemi scientifici e ingegneristici
384 pag. **L. 25.000**
Cod. 554P

Microprocessori e interfacciamento

USARE IL MICROPROCESSORE
L'utilizzo più razionale del microprocessore nel controllo di impianti e processi
296 pag. **L. 17.000**
Cod. 327A

MICROPROCESSORI
Dai Chip ai Sistemi.
I concetti, le tecniche e i componenti riguardanti il mondo dei microprocessori
384 pag. **L. 25.000**
Cod. 320P

I MICROPROCESSORI E LE LORO APPLICAZIONI: SC/MP
La soluzione dei classici problemi che si presentano nella progettazione con sistemi a microprocessore
158 pag. **L. 11.000**
Cod. 301D

Z80 PROGRAMMAZIONE IN LINGUAGGIO ASSEMBLY
Le funzioni assembler, le istruzioni assembly, i concetti di sviluppo del software
640 pag. **L. 34.000**
Cod. 326P



PROGRAMMAZIONE DELLO Z 80
Un corso sistematico per imparare la programmazione in linguaggio Assembler usando lo Z-80
530 pag. **L. 26.000**
Cod. 328D

NANOBOOK Z80
I nanocomputer NBZ80 e NBZ80S usati come strumenti didattici, per imparare il software, per affrontare i problemi e le tecniche di interfacciamento con CPU, PIO, CTC
Vol. I: Tecniche di Programmazione
256 pag. **L. 17.000**
Cod. 310P
Vol. III. Tecniche di interfacciamento
464 pag. **L. 20.000**
Cod. 312P

IL BUGBOOK VII
L'interfacciamento fra microcomputer e convertitori analogici. Esperimenti per sistemi 8080/A - Z80 - 8085
272 pag. **L. 17.000**
Cod. 007A

TECNICHE D'INTERFACCIAMENTO DEI MICROPROCESSORI
I concetti, le tecniche di base, i componenti per assemblare un sistema
400 pag. **L. 25.000**
Cod. 314P

LA PROGRAMMAZIONE DELLO Z 8000
L'architettura e il funzionamento, nonché molti esempi di programmi dello Z8000
302 pag. **L. 25.000**
Cod. 321D

PROGRAMMAZIONE DELLO Z80 E PROGETTAZIONE LOGICA
Linguaggio assembly e logica digitale, più alcune efficienti soluzioni per spiegare l'uso corretto del microprocessore
400 pag. **L. 21.500**
Cod. 324P

PROGRAMMAZIONE DEL 6502
Un testo autonomo e completo per imparare la programmazione in linguaggio Assembler
390 pag. **L. 25.000**
Cod. 503B

GIOCHI CON IL 6502
Tecniche di programmazione avanzate e loro sperimentazione attraverso il modo pratico e divertente dei giochi
312 pag. **L. 19.500**
Cod. 505B

ESPERIMENTI CON TTL E 8080A
Elettronica digitale, tecniche di programmazione e interfacciamento dei microcomputer
Vol. I
496 pag. **L. 22.000**
Cod. 005A
Vol. II
490 pag. **L. 22.000**
Cod. 006A

PROGRAMMAZIONE DELL'8080 E PROGETTAZIONE LOGICA
Implementazione della logica sequenziale e combinatoria, e uso del linguaggio assembly, all'interno di un sistema basato sull'8080
296 pag. **L. 19.000**
Cod. 325P

APPLICAZIONI DEL 6502
Le tecniche e i programmi per applicazioni tipiche del 6502 e dei sistemi su di lui basati
214 pag. **L. 15.500**
Cod. 504B

TEA, UN EDITOR ASSEMBLER RESIDENTE PER L'8080/8085
Un valido contributo per scrivere e modificare programmi sorgente scritti in assembler secondo i codici mnemonici dei due microprocessori
252 pag. **L. 14.000**
Cod. 322P

DBUG
Un programma interprete per la messa a punto del software 8080
112 pag. **L. 7.000**
Cod. 313P

8080A/8085 PROGRAMMAZIONE IN LINGUAGGIO ASSEMBLY
Un manuale teorico-pratico per tecnici, studenti, appassionati che vogliano approfondire le loro conoscenze nel settore dei microcomputer
512 pag. **L. 27.500**
Cod. 323P

INTERFACCIAMENTO DI MICROCOMPUTER
Esperimenti utilizzanti il CHIP 8255 PPI, interfaccia periferica programmabile della famiglia 8080
220 pag. **L. 12.000**
Cod. 004A



Elettronica

CORSO DI ELETTRONICA FONDAMENTALE
Un testo di alto valore didattico, per capire l'elettronica: dalla teoria atomica ai transistori
448 pag. **L. 17.000**
Cod. 201A

COMPRENDERE L'ELETTRONICA A STATO SOLIDO
12 lezioni complete ed esaurienti a cura del Learning Center Texas Instruments
224 pag. **L. 16.000**
Cod. 202A

CORSO PROGRAMMATO DI ELETTRONICA ED ELETTROTECNICA
In 40 fascicoli monografici, di 2700 pagine complessive, i concetti fondamentali di elettrotecnica ed elettronica di base: dalla teoria atomica all'elaborazione dei segnali digitali. 1000 lezioni con domande, risposte, esercizi, test...
L. 109.000 Cod. 099A

INTRODUZIONE AI CIRCUITI INTEGRATI DIGITALI
Un'introduzione pratica che demistifica molti luoghi comuni e rende accessibile a tutti l'argomento
112 pag. **L. 8.000**
Cod. 203A

ELETTRONICA INTEGRATA DIGITALE
Un testo didattico chiaro, completo, moderno, con oltre 400 problemi, dedicato a specialisti e studenti. Fondamentale
720 pag. **L. 38.000**
Cod. 204A

Componenti e progetti

DAL TRANSISTOR AL MICROPROCESSORE
La moderna circuteria a stato solido, la sua evoluzione, le sue prospettive
80 pag. **L. 7.500**
Cod. 141A

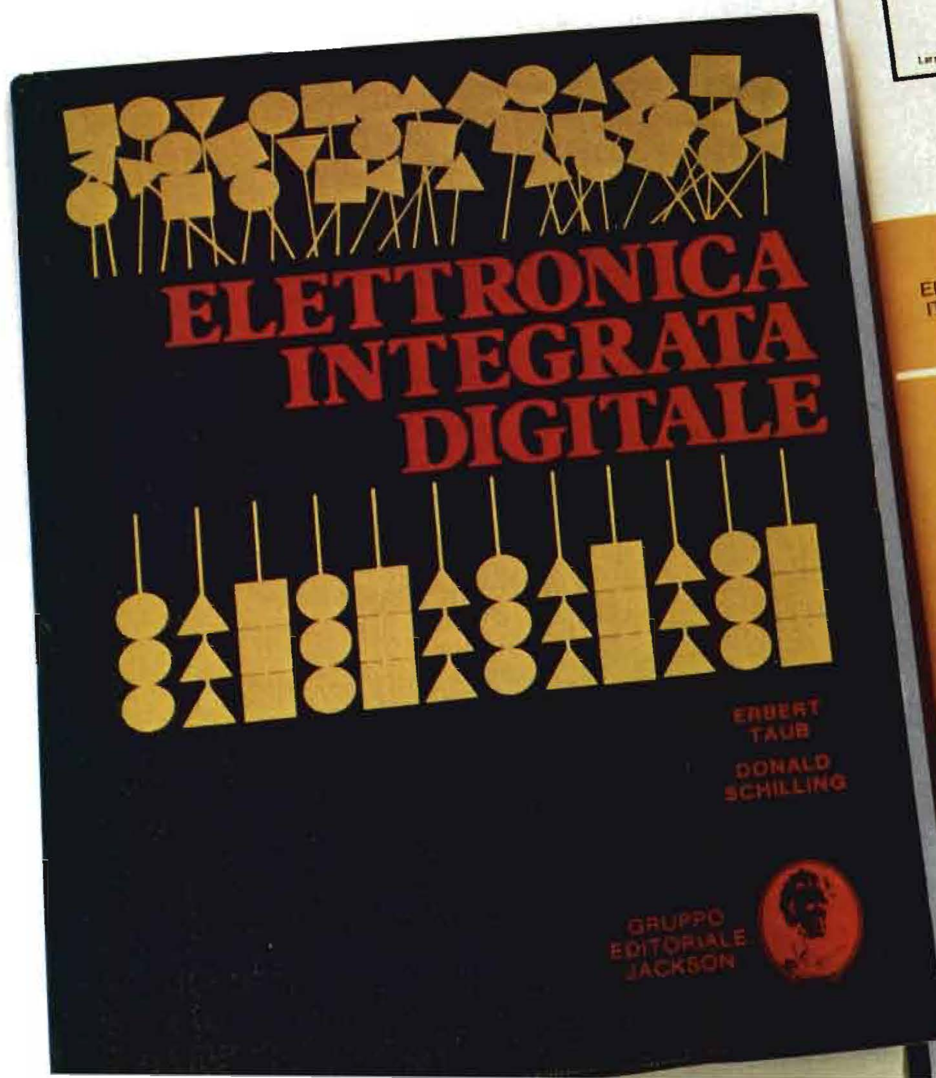
MANUALE PRATICO DI PROGETTAZIONE ELETTRONICA
Per hobbisti, dilettanti, sperimentatori e ingegneri alle prese con la comprensione e l'uso dei circuiti elettronici
488 pag. **L. 30.000**
Cod. 205A

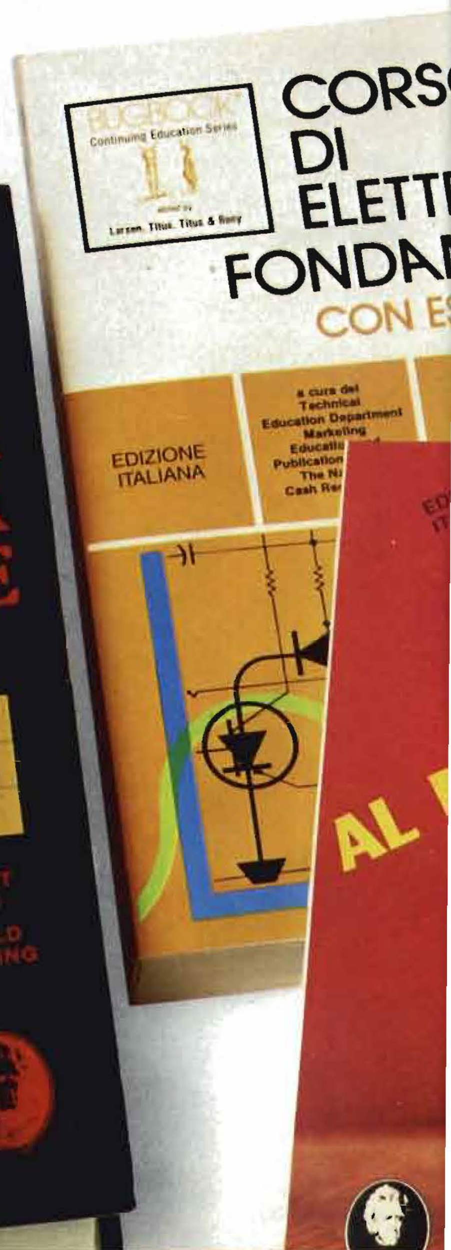
CIRCUITI LOGICI E DI MEMORIA
Un approccio diretto al mondo dell'elettronica digitale
Vol. I
384 pag. **L. 22.000**
Cod. 001A
Vol. II
352 pag. **L. 22.000**
Cod. 002A

I TIRISTORI 110 PROGETTI PRATICI
Dal controllo della luminosità delle lampade a quello (automatico) di stufe, dalla velocità dei motori elettrici ai sistemi antifurto...
144 pag. **L. 9.000**
Cod. 606D

MANUALE DEGLI SCR, TRIAC ED ALTRI TIRISTORI Vol. I
Una guida alle applicazioni di questa famiglia di dispositivi a semiconduttore
378 pag. **L. 24.000**
Cod. 612P

PROGETTAZIONE CIRCUITI PLL
L'oscillatore controllato in tensione, i sintetizzatori digitali di frequenza, i circuiti integrati monolitici...
256 pag. **L. 16.000**
Cod. 604H





LA PROGETTAZIONE DEI FILTRI ATTIVI
Attraverso una vasta gamma di tavole e grafici una pratica esemplificazione di come si costruiscono i filtri attivi
280 pag. **L. 17.000**
Cod. 603B

GUIDA AI CMOS
22 utili esperimenti per passare dalla logica TTL a quella CMOS
220 pag. **L. 17.000**
Cod. 605B

GLI AMPLIFICATORI DI NORTON QUADRUPLI LM 3900 E LM 359
Teoria, sperimentazione e... pratica attraverso 22 esperimenti realizzati passo passo
480 pag. **L. 24.000**
Cod. 610B

IL TIMER 555
Cos'è e come si utilizza questo onnipresente temporizzatore integrato
172 pag. **L. 10.000**
Cod. 601B

LA PROGETTAZIONE DEI CIRCUITI OP-AMP
Gli schemi di circuiti fondamentali che costituiscono le unità di base dei sistemi più sofisticati
276 pag. **L. 17.000**
Cod. 602B

GUIDA MONDIALE DEI TRANSISTORI
286 pag. **L. 23.000**
Cod. 607H

GUIDA MONDIALE DEGLI AMPLIFICATORI OPERAZIONALI
196 pag. **L. 17.000**
Cod. 608H

GUIDA MONDIALE DEI TRANSISTORI AD EFFETTO DI CAMPO JFET E MOS
80 pag. **L. 11.500**
Cod. 609H

LA SOPPRESSIONE DEI TRANSITORI DI TENSIONE
Cause, effetti, rimedi ai danneggiamenti, dei transitori d'alta tensione
224 pag. **L. 12.000**
Cod. 611H

Varie

LE COMUNICAZIONI RADIO IN MARE
Come orientarsi grazie alla moderna strumentazione e ai suoi codici
200 pag. **L. 15.000**
Cod. 706A

MANUALE PRATICO DEL RIPARATORE RADIO TV
I segreti di un'esperienza ventennale messi al servizio di tutti
352 pag. **L. 23.000**
Cod. 701P

IMPIEGO PRATICO DELL'OSCILLOSCOPIO
Come funziona e come usare —con facilità e precisione— questo indispensabile strumento
112 pag. **L. 16.000**
Cod. 705P

AUDIO & HI-FI
Una guida preziosa per conoscere l'HI-FI
128 pag. **L. 7.000**
Cod. 703D

MANUALE PRATICO DI REGISTRAZIONE MULTIPISTA
Regole generali, problemi, soluzioni e termini gergali
164 pag. **L. 10.000**
Cod. 704D





ATTENZIONE!

Per ordinare i libri presentati in queste pagine servirsi di questa Cedola di Commissione Libraria. Fino al 28/2/'84 a tutti gli Abbonati JACKSON viene riconosciuto uno sconto del 20% sui prezzi di copertina indicati. Dopo tale data gli Abbonati avranno sempre comunque diritto a uno sconto del 10% sulla Biblioteca JACKSON, novità comprese. I libri presentati possono essere ordinati a prezzo pieno con questa Cedola, anche dai non abbonati.

CEDOLA DI COMMISSIONE LIBRARIA

Spett. Gruppo Editoriale Jackson Divisione Libri
Sono interessato a ricevere i volumi sotto elencati alle condizioni di pagamento indicate.

Ritagliare e spedire in busta chiusa a: Gruppo Editoriale Jackson - via Rosellini 12 - 20124 Milano

Spazio riservato alle Aziende
☐ Si richiede l'emissione della fattura
Partita I.V.A. ☐☐☐☐☐☐☐☐☐☐☐

Cognome ____ Nome ____
Città ____ Prov. ____ C.A.P. ____
Via, ecc. ____ Nome della via, ecc. ____ Numero ____
Data di nascita ____ Telefono ____

NELLA COMPILAZIONE DELL'ORDINE INDICARE ESATTAMENTE CODICE E QUANTITÀ DEI VOLUMI

Cod. Vol.	Quant.	Cod. Vol.	Quant.	Cod. Vol.	Quant.	Cod. Vol.	Quant.	Cod. Vol.	Quant.	Cod. Vol.	Quant.	Cod. Vol.	Quant.

Contributo spese di spedizione L. 2000

☐ **Sono un vostro abbonato a**: ☐ l'Elettronica ☐ Elettronica Oggi ☐ Automazione/Oggi
☐ Elektor ☐ Informatica Oggi ☐ Video Giochi ☐ Personal Software ☐ Bit ☐ Telecomunicazioni/Oggi
☐ Strumenti Musicali E ho quindi diritto al 20% di sconto ☐ **Non sono abbonato**

☐ **DESIDERO PRENOTARE AL PREZZO SPECIALE DI SOTTOSCRIZIONE DI L. 200.000 *(COMPRESE SPESE DI SPEDIZIONE)* E.I. ENCICLOPEDIA DI ELETTRONICA E INFORMATICA.**

Modalità di pagamento:

☐ Allego assegno n° ____ Banca ____ Importo ____ Data __ / __ / __ Firma ____

☐ Ho già provveduto al pagamento con versamento su c/c n. 11666203 intestato a: Gruppo Editoriale Jackson *(allego fotocopia)*

☐ Ho già provveduto al pagamento con vaglia postale intestato a: Gruppo Editoriale Jackson *(allego fotocopia)*

☐ Pagherò al postino al ricevimento dei volumi (modalità non valida per l'enciclopedia)

Giochi di inseguimento

Strategie di caccia e di programmazione

di *Marcello Morchio*

In molti video game di movimento il computer cerca di mangiare il giocatore. Marcello Morchio di Genova ci ha suggerito tre diverse strategie di caccia in un gioco per lo ZX81 che contiene anche alcuni trucchi da tenere sempre presenti.

Si gioca in un campo diviso in tre scomparti da dei muri, che non possiamo oltrepassare pena la distruzione. Nella situazione iniziale noi siamo l'asterisco nello scomparto a sinistra del campo mentre l'inseguitore è la croce a destra. Lui comincia subito a venire verso di noi, aprendo dei varchi nei muri: sarà proprio da quei varchi che dovremo cercare di passare per poter sopravvivere. Scopo del gioco è resistere il più a lungo possibile.

Il movimento si ottiene utilizzando i tasti con le frecce, ma modificando le linee 120 e 130 lo si può ottenere con qualunque altra combinazione di tasti.

Le linee fino alla 95, eseguite in modo fast, inizializzano il gioco. Dalla linea 100 alla 210 c'è il ciclo di movimento e controllo, eseguito in modo slow per avere la permanenza dell'immagine, dalla 220 in poi la conclusione con la stampa del punteggio.

Il programma usa poche variabili. A e B sono l'ascissa e l'ordinata dell'inseguitore (attenzione che l'origine degli assi è la posizione di 0, 0 in alto a sinistra), X e Y sono l'ascissa e l'ordinata del giocatore (simbolo "*"), P è il punteggio e H è il parametro casuale che determina se l'inseguimento deve avvenire in linea orizzontale o verticale.

La prima cosa da notare è l'uso intelligente delle funzioni logiche del BASIC Sinclair. Le linee 120 e 130 sono equivalenti a:

```
120 IF A$="6" THEN LET X=X+1
125 IF A$="7" THEN LET X=X-1
130 IF A$="8" THEN LET X=Y+1
135 IF A$="5" THEN LET Y=Y-1
```

Infatti l'espressione, per es., (A$ = "6") vale 1 se A$ è "6", mentre vale zero in tutti gli altri casi, quindi sarà X = X + 1 se A$ = "6" e X = X − 1 se A$ = "7". Si noti che sui Sinclair la condizione logica VERO vale 1 e non −1 come in molti altri dialetti BASIC. Lo stesso tipo di espressione compare alle linee 150 e 160 che calcolano le nuove coordinate dell'inseguitore.

L'inseguimento può essere reso più incalzante in due modi: modificando le linee 150 e 160 così:

```
150 IF H=0 OR B=Y THEN
    LET A=A+(X>A) - (X<A)
160 IF H=1 OR A=X THEN
    LET B=B+(Y>B) - (Y<B)
```

l'inseguitore si comporta come prima quando si trova in diagonale rispetto al giocatore, ma quando si trova sulla stessa ascissa o sulla stessa ordinata il parametro H viene neutralizzato dalla seconda condi-

zione delle IF. Ciò causa un raddoppio nella velocità di inseguimento che impedisce al giocatore di aumentare le distanze.

Se si eliminano le condizioni, così:

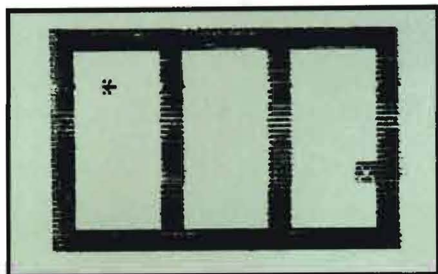


Figura 1. *Hardcopy del quadro iniziale.*

```
150 LET A=A+(X>A) - (X<A)
160 LET B=B+(Y>B) - (Y<B)
```

l'inseguitore può muoversi anche in diagonale, cosa che il giocatore non può fare perché non può schiacciare contemporaneamente due tasti, e la fuga diventa impossibile.

Provate invece a cambiare nel programma originale la linea 140 con

```
140 LET H=INT (RND*4)
```

e osservate cosa succede.

Un altro punto importante è quello nel quale si controlla se il giocatore è andato a sbattere contro un muro o è stato preso dall'inseguitore.

Ci sono vari metodi per ottenere l'effetto: il primo si basa su semplici PEEK, cioè va a leggere nella mappa dello schermo dove dovrà essere scritto il "*" controllando se andrà contro un muro o sarà mangiato.

Il secondo metodo consiste in una routine in linguaggio macchina che svolge la stessa funzione.

Ecco il primo metodo:

PRINT AT X, Y; "questa istruzione punta la posizione X, Y"

IF PEEK (PEEK 16398 + 256 * PEEK 16399) <> 0 THEN GOTO 220

"220 è la fine del programma, dove viene mostrato il punteggio"

Ecco invece il listato della routine:

dec.	exdec.	assembler
42 14 64	2A 0E 40	LD HL, (16398)
78	4E	LD C, (HL)
6 0	06 00	LD B, 00
201	C9	RET

e che può essere caricato con questo programma:

```
1 REM (sette caratteri)
10 FOR K = 16514 TO 16520
20 INPUT A
30 POKE K, A
40 NEXT K
```

Bisogna inserire i codici DECIMALI ad ogni INPUT; poi cancellare le linee da 10 a 40 e poi scrivere il programma. Per mandarlo in esecuzione bisogna scrivere come per il primo metodo, solo che al posto dello IF PEEK (... bisogna scrivere LET I = USR 16514 e poi IF I <> 0 THEN GOTO 220. Questo secondo metodo è più veloce del primo, ma è un po' più complesso.

Si incoraggia il lettore a provare altre strategie di inseguimento e a modificare ed espandere il gioco a piacere. I possessori di Spectrum possono facilmente adattare questo programma per muovere un solo pixel in un campo molto più grande. Questo può essere il punto di partenza per ottenere il vostro gioco di inseguimento, con tanto di handicap, bonus, e... ■

```
   5 FAST
  10 PRINT "■■■■■■■■■■■■■■■■■■■■"
  20 FOR C=1 TO 8
  30 PRINT "■     ■     ■     ■"
  40 NEXT C
  50 PRINT "■■■■■■■■■■■■■■■■■■■■"
  60 LET A=7
  70 LET B=14
  80 LET X=2
  90 LET Y=2
  93 LET P=0
  95 SLOW
 100 PRINT AT X,Y;" ";AT A,B;" "
 110 LET A$=INKEY$
 120 LET X=X+(A$="6")-(A$="7")
 130 LET Y=Y+(A$="8")-(A$="5")
 140 LET H=INT (RND*2)
 150 IF H=0 THEN LET A=A+(X>A)-(
X<A)
 160 IF H=1 THEN LET B=B+(Y>B)-(
Y<B)
 170 PRINT AT A,B;"■";AT X,Y;
 180 IF PEEK (PEEK 16398+256*PEE
K 16399)<>0 THEN GOTO 220
 190 PRINT "*"
 200 LET P=P+1
 210 GOTO 100
 220 PRINT AT X,Y;"■",P
 230 PAUSE 4E4
 240 CLS
 250 RUN
```

Nota:
nelle linee 10, 30, 50, il carattere "■" è uno spazio inverso, quello della linea 170 una "X" inversa; quello della linea 220 un asterisco inverso.

Listato 1. *Listato normale.*

```
   1 REM E:RND?▞ TAN
   5 FAST
  10 PRINT "■■■■■■■■■■■■■■■■■■■■"
  20 FOR C=1 TO 8
  30 PRINT "■     ■     ■     ■"
  40 NEXT C
  50 PRINT "■■■■■■■■■■■■■■■■■■■■"
  60 LET A=7
  70 LET B=14
  80 LET X=2
  90 LET Y=2
  93 LET P=0
  95 SLOW
 100 PRINT AT X,Y;" ";AT A,B;" "
 110 LET A$=INKEY$
 120 LET X=X+(A$="6")-(A$="7")
 130 LET Y=Y+(A$="8")-(A$="5")
 140 LET H=INT (RND*2)
 150 IF H=0 THEN LET A=A+(X>A)-(
X<A)
 160 IF H=1 THEN LET B=B+(Y>B)-(
Y<B)
 170 PRINT AT A,B;"■";AT X,Y;
 180 LET I=USR 16514
 185 IF I<>0 THEN GOTO 220
 190 PRINT "*"
 200 LET P=P+1
 210 GOTO 100
 220 PRINT AT X,Y;"■",P
 230 PAUSE 4E4
 240 CLS
 250 RUN
 300 FOR K=16514 TO 16520
 310 INPUT A
 320 POKE K,A
 330 NEXT K
```

Nota:
nelle linee 10, 30, 50, il carattere "■" è uno spazio inverso, quello della linea 170 una "X" inversa; quello della linea 220 un asterisco inverso.

Listato 2. *Listato con routine in linguaggio macchina già caricata e caricatore attivabile con RUN 300.*

Area per linguaggio macchina nel BASIC ZX81

Qualche utile suggerimento per i vostri programmi

di *Michele Petraccone*

Per risparmiare memoria e tempi di comunicazione con il registratore, è molto utile inserire dati o programmi in linguaggio macchina, con valori direttamente usabili, in linee REM del programma BASIC, generalmente al suo inizio per (ovvi) motivi di stabilità degli indirizzi.

Quando l'area di memoria da riservare è piuttosto estesa, diventa molto noioso scrivere la lunga REM necessaria. Si può anche dividerla in più linee, ma questa comodità si paga poi in termini di occupazione di memoria (6 byte per ogni linea) e soprattutto in termini di flessibilità, in quanto modificando i programmi LM si deve tener conto della lunghezza della linea.

Ecco quindi una ricetta per sveltire la preparazione di una linea REM, lunga quanto si vuole.

Scrivete una linea di questo tipo:

```
1 REM 000000000X000000000X000
000000X000000000X0000
```

Nella linea sono compresi 44 caratteri; la sua lunghezza effettiva è di 50 byte = 2 (numero linea) + 2 (lungh. linea) + 1 (REM) + 44 + 1 (118 = separatore di linea), ma la lunghezza indicata in 16511/12 è 46 = 1 (REM) + 44 + 1 (118), cioè l'intera lunghezza meno le due coppie di byte iniziali.

Con la funzione EDIT ricopiate tante linee uguali alla prima quante ve ne servono.

Poniamo che abbiate ricopiato altre 9 linee; in tal caso avrete occupato 10 × 50 = 500 byte, dal 16509 al 17008. Adesso vorremmo che il BASIC non utilizzasse questa zona, ma che nella esecuzione dei programmi partisse direttamente da 17009: basta simulare che la prima linea abbia una lunghezza pari all'intera zona che ci interessa riservare.

Calcoliamo la nuova lunghezza:

(500−4): 256 = 1 (MSB)
con resto 240 (LSB).

Attribuiamo questi nuovi valori con:

```
POKE 16511, 240
POKE 16512, 1
```

Per serrare le linee sullo schermo dovreste eliminare i 118 a partire da 16558 fino a 16958, con passi di 50, mediante istruzioni immediate (POKE 16558, 0; POKE 16608, 0; ....; POKE 16958, 0).

Per chi vuol far eseguire automaticamente la cancellazione dei 118 ed ottenere anche una maggiore flessibilità, consiglio il seguente metodo:

```
1 REM
000000000X000000000X000
000000X0000000000X0000
                    ('50 16558')
 2 REM
 ...
 (da 2 a 8 altre 7 linee copiate dalla 1)
 ...
```

```
8 REM                  ( 50 16908 )
70 FOR K = 16514
   TO 17045            ( 31 16939 )
72 IF PEEK K = 118
   THEN POKE K, 0      ( 29 16968 )
74 NEXT K              (  7 16975 )
76 POKE 16511, 037     ( 27 17002 )
78 POKE 16512, 002     ( 27 17029 )
80 REM 000000000X      ( 16 17045 )
82 REM 000000000X      ( 16 17061 )
```

I numeri tra parentesi indicano per ogni linea la lunghezza e l'indirizzo del 118 finale.

Nei valori POKE delle linee 76 e 78 sono scritti anche gli zeri non significativi per avere sempre un totale di 3+3 byte, anche se varia il valore rappresentato.

Date il RUN. Dopo cinque secondi il programma avrà inglobato nella prima le linee da 2 a 82, e avrete pronti 547 byte disponibili da 16514 a 17060.

Si faccia attenzione a non trattare la linea 1 con EDIT altrimenti il sistema operativo dello ZX81 incontrerebbe in una linea REM dei 126, che sono i separatori tra la codificazione decimale e quella esponenziale, in 5 byte, delle variabili alle linee da 70 a 78. Il risultato sarebbe l'eliminazione dei 126 e dei 5 byte successivi.

Si noti che una routine in linguaggio macchina difficilmente non conterrà un byte a 126 (7E) che corrisponde all'istruzione LD A, (HL), molto usata.

Le due linee REM 80 e 82 servono per accorciare l'area riservata, senza usare EDIT.

Per accorciarla di 32 byte, basta eliminare le due linee, eseguendo in modo immediato:

```
POKE 17045, 118
POKE 17029, 118
POKE 16511, 5
```

e cancellandole.

Allungare è più facile; per aggiungere 32 byte, scrivete:

```
90 REM... (26 caratteri)...
```

poi, in modo immediato:

```
POKE 16511, 69
POKE 17061, 0
```

Alle linee del programma sono stati attribuiti i numeri da 70 a 82 e non numeri più bassi, perché in tal caso il sistema operativo, già privo del riferimento dei 118, si confonderebbe bloccandosi sulla linea NEXT K con errore C. ■

Accesso per chiave logica ad un archivio "relative"

Una efficiente tecnica di memorizzazione dati

di *Gioele Confortini*

L'utilizzo di un archivio "relative" o, come definito in BASIC, "random" offre indubbi vantaggi sia per rapidità d'accesso sia per il minimo impegno di memoria richiesto dal software necessario alla gestione dell'archivio.

L'organizzazione relative, però, offre il fianco ad una critica: l'accesso è limitato alla conoscenza "a priori" del numero relativo di record che si vuol trattare costituendo tale numero, contemporaneamente, l'unica chiave identificativa del record stesso.

Da queste due limitazioni nascono, da una parte, problemi di sviluppo di applicazioni e, dall'altra, se l'applicazione non è stata studiata con una certa cura, difficoltà o limitatezza di utilizzo della stessa.

Mantenere, poi, fino in fondo tutti i vantaggi dell'organizzazione "relative" e, contemporaneamente, la flessibilità sia di sviluppo che di utilizzo dell'accesso per chiave logica è un po' come avere la botte piena e la moglie ubriaca: qualcosa bisogna concedere (o alla botte o alla moglie) ma se il risultato sarà ragionevolmente accettabile il peso dell'implementazione non sarà perso.

L'utilizzo di strutture concettuali collaudate (liste ed alberi binari) ha permesso una soluzione semplice del problema: non si fa riferimento ad alcun linguaggio di programmazione anche se la presenza di algoritmi ricorsivi suggeriscono implicitamente il Pascal per l'eleganza formale che permette.

Di ogni problema affrontato si fornisce il diagramma a blocchi risolutivo ed un breve commento dello stesso, nella convinzione che "una rappresentazione grafica vale migliaia di parole" (con il che, mentre si segue la tendenza attuale dell'informatica, si rivaluta un po' la funzione del tanto bistrattato diagramma a blocchi).

Tutta la discussione è relativa ad una sola chiave logica: la generalizzazione a più chiavi non presenta particolari problemi.

Archivio indici

È un archivio di supporto mediante il quale è pilotato l'accesso all'archivio di lavoro: ha struttura "relative".

I record che compongono l'archivio indici sono:

N° record	Contenuto
1	R(elative) R(ecord) N(umber) del primo record disponibile sull'archivio di lavoro;
2	– identificazione della prima chiave logica dell'archivio di lavoro, – caratteristiche della prima chiave logica: formato, lunghezza, posizione nel record dell'archivio di lavoro, – RRN del primo re-

cord in sequenza ascendente per questa chiave = forward link radice,
- RRN del primo record in sequenza discendente per questa chiave = backward link radice,
- medium pointer di partenza dell'albero binario dell'archivio di lavoro per questa chiave;

3÷n come il record N° 2 ma relativi alla 3ª ÷ ennesima chiave logica.

Accesso sequenziale per chiave logica

Richiede che venga costruita una lista semplice: per completezza la lista viene sviluppata a doppio legame ma, se conveniente, il backward link può essere tralasciato.

La funzione torna utile in tutti quei casi nei quali è necessaria una elencazione in ordine di chiave logica: nomi, descrizioni, codici di lavoro (provincia, rappresentante, sindacato, CAP, ecc.).

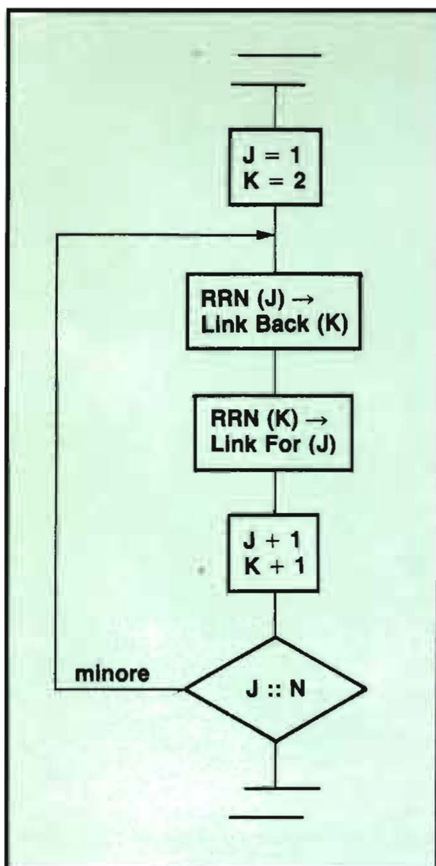


Figura 2. *Sviluppo della lista.*

Figura 1. *Esempio numerico.*

N° 1 N = 8

	sviluppate					corrette	
colonne	1	2	3	4	5	4	5
riga 1	1	4	8	2	3	2	3
riga 2	1	2	4	0	4	7	4
riga 3	4	6	8	5	6	5	6
riga 4	2	3	4	0	0	0	0
riga 5	4	5	6	0	0	0	0
riga 6	6	7	8	0	0	0	8
riga 7	0	1	0	0	0	0	0
riga 8	0	8	0	0	0	0	0

N° 2 N = 15

	sviluppate					corrette	
colonne	1	2	3	4	5	4	5
riga 1	1	8	15	2	3	2	3
riga 2	1	4	8	4	5	4	5
riga 3	8	11	15	6	7	6	7
riga 4	1	2	4	0	8	14	8
riga 5	4	6	8	9	10	9	10
riga 6	8	9	11	0	11	0	11
riga 7	11	13	15	12	13	12	13
riga 8	2	3	4	0	0	0	0
riga 9	4	5	6	0	0	0	0
riga 10	6	7	8	0	0	0	0
riga 11	9	10	11	0	0	0	0
riga 12	11	12	13	0	0	0	0
riga 13	13	14	15	0	0	0	15
riga 14	0	1	0	0	0	0	0
riga 15	0	15	0	0	0	0	0

I passi da programmare sono:

1) Caricamento in memoria di una tabella di lavoro dall'archivio dati.
   Ogni elemento di tabella è costituito da:
   a) RRN del record letto,
   b) forward link (inizializzato a zero),
   c) backward link (inizializzato a zero),
   d) flag di lavoro (inizializzato a zero),
   e) left pointer (inizializzato a zero),
   f) right pointer (inizializzato a zero),
   g) chiave di cui si vuole l'ordinamento.
   Sviluppare un contatore N che contenga il numero degli elementi della tabella.
2) Ordinamento della tabella per chiave.
3) Sviluppo della lista (figura 2): comporta che:
   - il RRN del 2° elemento di tabella sia portato sul F.L. del 1° elemento di tabella,
   - il RRN del 1° elemento di tabella sia portato sul B.L. del 2° elemento di tabella,
   - il procedimento venga iterato sino all'esaurimento del numero degli elementi della tabella.
   Al termine del ciclo, il B.L. del 1° elemento della tabella e il F.L. dell'ultimo elemento della tabella sono a zero: in fase di scansione dell'archivio tale valore indica il termine dell'operazione di scansione.
   Al termine del ciclo, il RRN del 1° elemento verrà riportato sul F.L. radice del record indice per questa chiave: il RRN dell'ultimo elemento verrà riportato sul B.L. radice del record indice per questa chiave.

Accesso random per chiave logica

La presentazione sequenziale di un archivio è un fatto che nella pratica si verifica con una frequenza piuttosto bassa: produzione di listini, liste alfabetiche di vario tipo, elenchi per provincia ecc. non sono elaborazioni giornaliere mentre, di contro, l'interrogazione estemporanea per conoscere qualche cosa è all'ordine del minuto.

Per accelerare i tempi di ricerca si è adottata una struttura ad albero binario: ma un albero binario, per poter funzionare correttamente, richiede che non vi siano chiavi doppie. È necessario, pertanto, esaminare e correggere la tabella costruita per eliminare (ma non perdere!) le eventuali chiavi doppie presenti.

I passi da programmare sono:

1) Identificazione delle chiavi doppie (figura 3):
   se due o più chiavi risultano uguali, il relativo flag di lavoro viene posto uguale a 1.
   La scansione viene fatta su tutti gli elementi della tabella.

2) Compattazione della tabella (figura 4): questo comporta che:
   - gli elementi dei vari sottoinsiemi della tabella identificati dal flag di lavoro = 1 (escluso il primo di ogni sottoinsieme) siano aggiornati sull'archivio dati,
   - gli elementi dei vari sottoinsiemi della tabella identificati dal flag di lavoro = 1 (escluso il primo di ogni sottoinsieme) siano cancellati dalla tabella spostando verso l'alto i restanti elementi della tabella,
   - il valore di N sia corretto del numero degli elementi cancellati.

3) Sviluppo dell'albero binario (figura 5):
   si calcola la media delle coppie di valori minimo-medio/medio-massimo ottenuti dal calcolo precedente,
   le terne sviluppate (minimo-medio-massimo) vengono memorizzate solo se i valori minimo/medio sono diversi: in questo caso, viene memorizzato anche il poin-

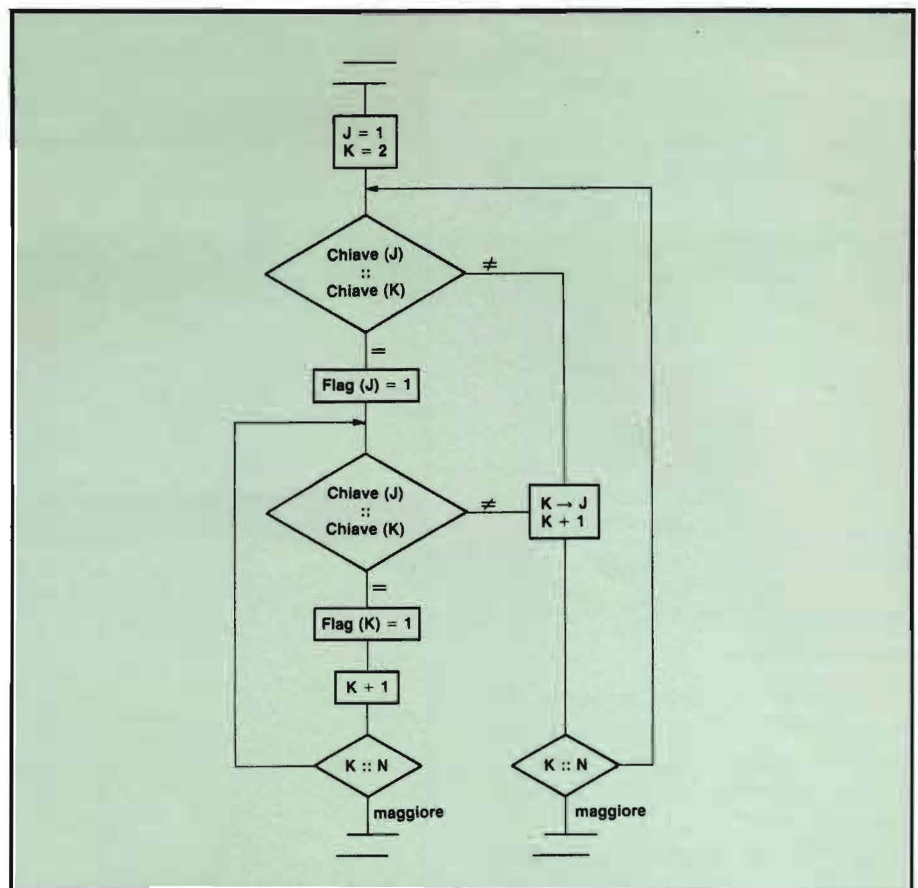


Figura 3. *Identificazione delle chiavi doppie.*

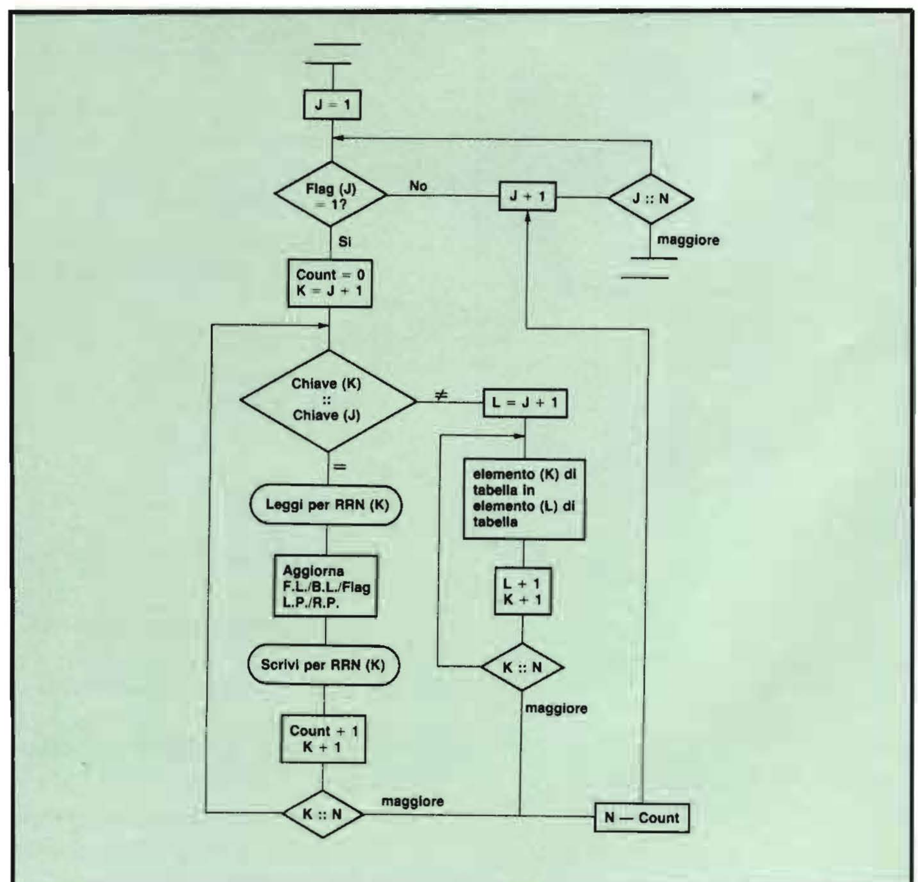


Figura 4. *Compattazione della tabella.*

L = 1
R = N

$M = \frac{L + R}{2}$

A (1,1) = L
A (1,2) = M
A (1,3) = R

J = 1
K = 2

L = A (J, 1)
R = A (J, 2)

$M = \frac{L + R}{2}$

L = M?
Si
No

A (K, 1) = L
A (K, 2) = M
A (K, 3) = R

A (J, 4) = K
K + 1

L = A (J, 2)
R = A (J, 3)

$M = \frac{L + R}{2}$

L = M?
Si
No

A (K, 1) = L
A (K, 2) = M
A (K, 3) = R

A (J, 5) = K
K + 1

J + 1

J < K?
Si
No

A (K, 2) = 1
A (K + 1, 2) = N
A (K − 1, 5) = N
J = 1

A (J, 2) = 2?
No
J + 1

A (J, 4) = K

Figura 5. *Sviluppo dell'albero binario sulla matrice A (N, 5).*

ter della riga sulla quale la terna è archiviata.
Al termine del ciclo, vengono memorizzati anche i valori minimo e massimo di partenza.
Tutta l'elaborazione si sviluppa su di una matrice di N righe per 5 colonne: il valore di N è quello corretto al punto 2) precedente.

Come leggere la matrice (vedi esempio numerico):
le colonne significative sono la 2), la 4) e la 5).
La colonna 2) contiene il pointer all'elemento della tabella ordinata che, per quella riga, risulta essere il medio tra gli estremi (indicati in col. 1 e col. 3).
La colonna 4), se diversa da zero, indica la riga della matrice che, a colonna 2), contiene il predecessore dell'elemento medio.
La colonna 5), se diversa da zero, indica la riga della matrice che, a colonna 2), contiene il successore dell'elemento medio.
Se la colonna 4) è zero, il medio di colonna 2) non ha predecessori.
Se la colonna 5) è zero, il medio della colonna 2) non ha successori.
Se la colonna 4) e la colonna 5) sono entrambe a zero, il medio della colonna 2) è un elemento terminale dell'albero, cioè una foglia.

4) Trasposizione dell'albero binario sulla tabella (figura 6).
   Su ogni elemento di tabella, identificato da col. 2), deve essere trasferito nel left pointer il RRN dell'elemento di tabella puntato dal medio della riga di matrice identificata da col. 4), nel right pointer il RRN dell'elemento di tabella puntato dal medio della riga di matrice identificata da col. 5): a condizione, ovviamente, che né col. 4) né col. 5) siano a zero.
   Al termine del ciclo, il valore medio della prima riga di matrice va impostato sul medium pointer del record indici della chiave in elaborazione.

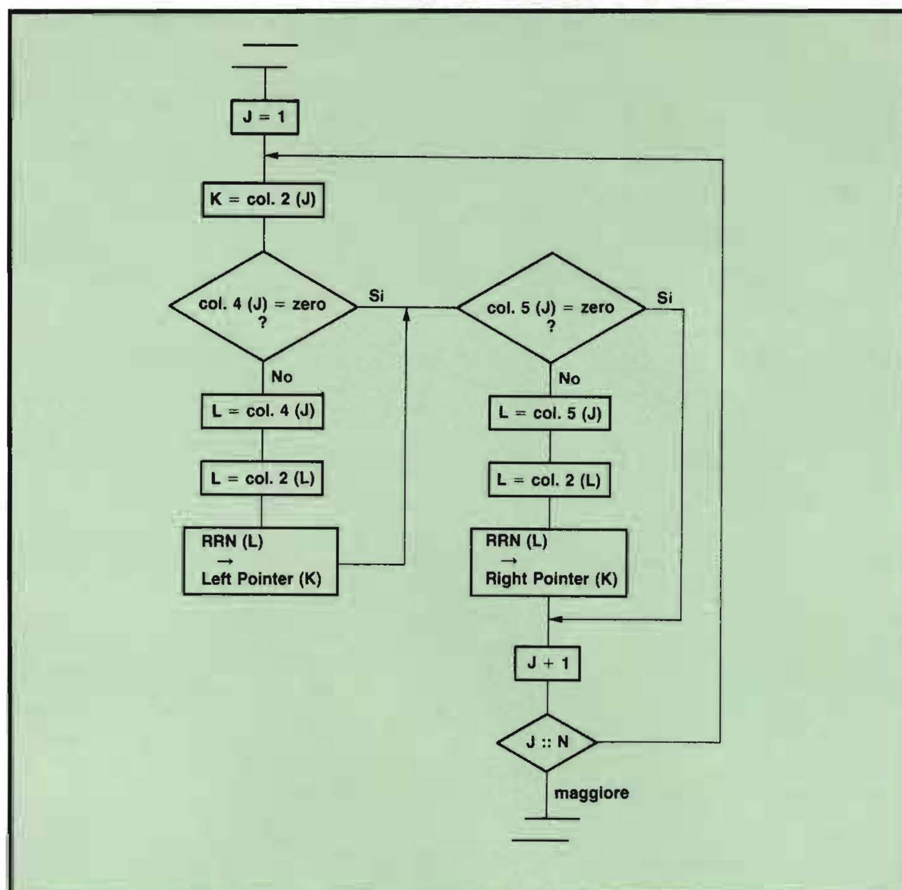


Figura 6. *Trasposizione dell'albero binario.*

Interrogazione dell'archivio

Ottenuta la chiave logica d'accesso e identificato e letto il corrispondente record indici, impostare come RRN di lettura il medium pointer: eseguita la lettura, confrontare la chiave data con la chiave letta.

a) Chiave data maggiore di chiave letta: se il right pointer non è zero, impostare come RRN di lettura il right pointer e ritornare alla lettura; se il right pointer è zero, si verifica la condizione di non trovato;

b) chiave data minore di chiave letta: se il left pointer non è zero, impostare come RRN di lettura il left pointer e ritornare alla lettura; se il left pointer è zero, si verifica la condizione di non trovato;

c) chiave data uguale a chiave letta: si verifica una condizione di incertezza in quanto potrebbero esservi più chiavi uguali: analizzare il flag di lavoro letto; se è zero, la chiave fornita è unica e si ha la condizione di trovato; se è 1, utilizzando il forward link, esporre i dati delle varie chiavi uguali sino ad avere conferma di quella richiesta.

Valutazione dei tempi

Considerando un archivio di 1,024 madri/figlie (2^{10}), si ha un massimo di 10 seek per individuare qualunque record: i tempi medi di accesso degli attuali floppy vanno da 70 a 100 millisecondi.

Quanto sopra porta a dire che, per l'archivio considerato, si ha un massimo d'attesa che varia tra 7/10 di secondo ed un secondo prima di avere la risposta: mediamente poi tali tempi si dimezzano.

Riduzione dei tempi di risposta

Anziché sviluppare l'albero binario sull'intera tabella ordinata, si può pensare di svilupparlo su frazioni della stessa: risulterà meno profondo e di conseguenza la ricerca sarà più rapida.

Questo porta a frazionare anche la chiave logica: non esistono condizioni particolarmente complesse in quanto può essere automaticamente definita in un campo di valori. Ad ogni limite superiore di valore corrisponderà un record indici.

Inserimenti (figura 7)

"Dove" inserire nell'archivio il nuovo record, è pilotato dal RRN presente sul 1° record dell'archivio indici: non presenta quindi particolari difficoltà.

L'operazione di inserimento diventa, invece, particolarmente delicata quando si passa a considerare le chiavi logiche presenti nel record: è necessario infatti aggiornare il left e il right pointer dell'albero binario nonché il forward ed il backward link della lista associati alle singole chiavi logiche. Inoltre, può verificarsi la condizione di "underflow" (inserire il minore del minimo) o di "overflow" (inserire il maggiore del massimo): e per lo stesso record, su chiavi logiche diverse, le due condizioni possono verificarsi.

Gli inserimenti vanno seguiti con una certa attenzione in quanto potrebbero portare ad una crescita sbilanciata dell'albero, con conseguente aumento dei tempi di ricerca: quando rigenerare l'albero potrebbe essere determinato o a scadenze fisse o in base al numero degli inserimenti.

Cancellazioni

Realizzare la cancellazione di un record inserendolo, ad esempio, in una catena di record disponibili può portare all'eliminazione di un nodo dell'albero: di conseguenza tutti i record dipendenti dal nodo cancellato sparirebbero a loro volta.

È più prudente, e più semplice, mantenere un flag di cancellazione su ogni record e, in sede di ristrutturazione dell'albero, scartare i record che abbiano il flag in on.

Bibliografia

Robert W. Elmore, Krishna K. Agarwal, "An Information - Retrieval System", Byte Ottobre 1980.
Ted Carter, "Implementing Dynamic Data Structures with BASIC file", Byte Febbraio 1980.
Mark Pelczarski, "Un data base modulare per l'Apple", **Personal Software** Giugno 1983. ■

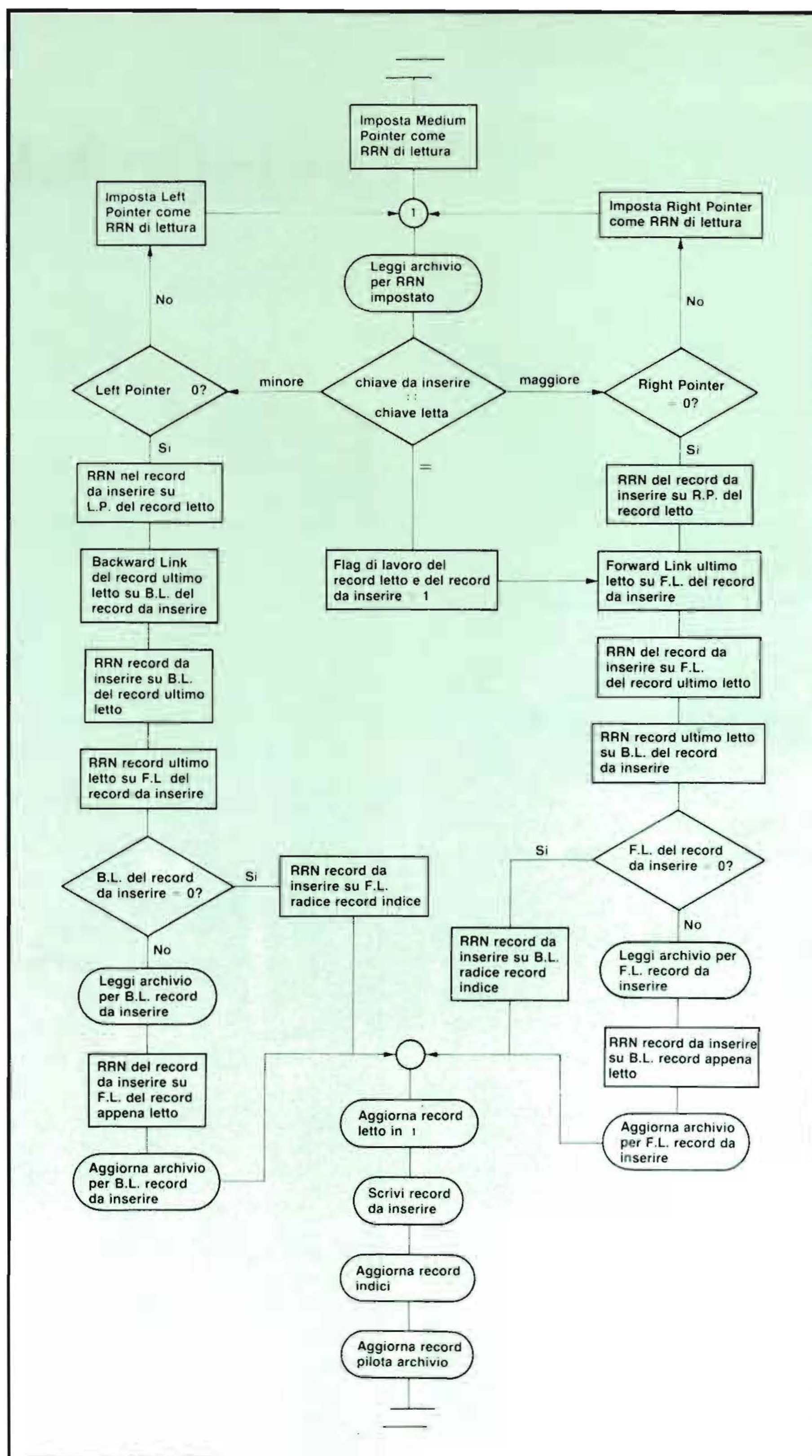


Figura 7. *Diagramma a blocchi della routine Inserimento.*

Dedalo 3-D

Guai a perdere l'orientamento. Dal labirinto non si esce più

di *Alessandro Guida*

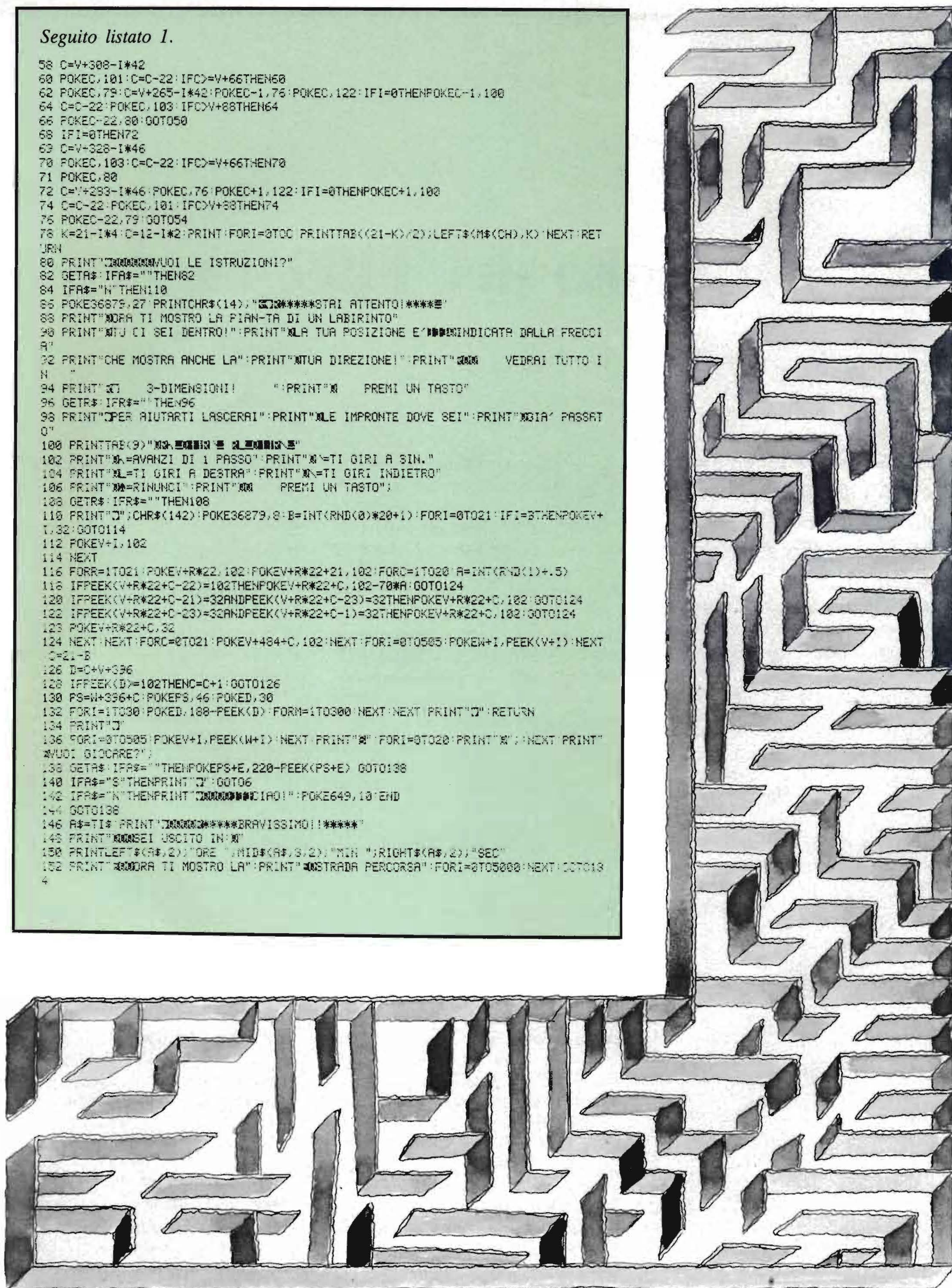
Il gioco consiste nel riuscire a venir fuori da un labirinto. Dopo le istruzioni viene mostrata la pianta del labirinto vista dall'alto, con una freccia che indica la posizione di partenza. Iniziato il gioco il labirinto viene visualizzato in tre dimensioni. Quindi, risulta visibile solo una parte del corridoio in cui ci si trova. Guai a perdere l'orientamento! Il programma gira sul VIC senza espansioni, o con l'espansione da 3 Kbyte. Con espansioni maggiori di 3 Kbyte occorre apportare le seguenti modifiche:

2 POKE 56, PEEK (56) –2: POKE 52, PEEK (52) –2
4 DIMM$(1),A$(3): V=4096: W= 15872 – 8192 ☆ (FRE(0) > 13000): E = V – W.

Attenzione che alcune linee di programma sono più lunghe di 80 caratteri per cui è necessario digitarle utilizzando le forme abbreviate dei comandi (per esempio, PRINT=?). Buon divertimento. ■

Listato 1. *Il programma Dedalo 3-D.*

```
1 REM*****************>DEDALO 3-D<************
2 POKE56,28:POKE55,5:POKE52,28:POKE51,5:POKE649,1
4 DIMM$(1),A$(3):V=7680:W=7174:E=506
6 GOSUB60:TI$="000000":M$(0)="░░░░░░░░░░░░░░░░░░░░░░":M$(1)="▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚"
8 A$(0)="ILMJ":A$(1)="JILM":A$(2)="MJIL":A$(3)="LMJI":B=0:SP=-22:SO=-1:GOTO36
10 PRINT"■■"TAB(7);TI$
12 GETM$:IFM$=""THEN10
14 IFM$="A"THEN134
16 FORI=1TO4:IFMID$(A$(B),I,1)<>M$THENNEXT
18 ONIGOTO20,22,24,26,28
20 B=0:SP=-22:SO=-1:GOTO30
22 B=1:SP=1:SO=-22:GOTO30
24 B=2:SP=22:SO=1:GOTO30
26 B=3:SP=-1:SO=22:GOTO30
28 GOTO12
30 IFM$<>"I"THEN36
32 IFPEEK(PS+SP)=102THEN12
34 IFPS+SP<W+22THEN146
35 PS=PS+SP:POKEPS,46
36 PRINT"■■";TAB(7)TI$:FORI=0TO20:POKEV+44+I,99:NEXT
38 FORI=0TO4
40 IFI>0ANDPEEK(PS+I*SP)=46THENPOKEV+318-I*44,42
42 CH=0:IFPS+I*SP<W+22ANDPEEK(PS+I*SP)=32THENCH=1:GOSUB78:GOTO12
44 IFPEEK(PS+I*SP)=102THENGOSUB78:GOTO12
46 IFPEEK(PS+I*SP+SO)<>102THEN56
48 POKEV+308-I*42,78:POKEV+287-I*42,78
50 IFPEEK(PS+I*SP-SO)<>102THEN68
52 POKEV+328-I*46,77:POKEV+305-I*46,77
54 NEXT:GOSUB78:GOTO12
56 IFI=0THEN62
```



Seguito listato 1.

```
58 C=V+308-I*42
60 POKEC,101:C=C-22:IFC>=V+66THEN60
62 POKEC,79:C=V+265-I*42:POKEC-1,76:POKEC,122:IFI=0THENPOKEC-1,100
64 C=C-22:POKEC,103:IFC>V+88THEN64
66 POKEC-22,80:GOTO50
68 IFI=0THEN72
69 C=V+328-I*46
70 POKEC,103:C=C-22:IFC>=V+66THEN70
71 POKEC,80
72 C=V+293-I*46:POKEC,76:POKEC+1,122:IFI=0THENPOKEC+1,100
74 C=C-22:POKEC,101:IFC>V+88THEN74
76 POKEC-22,79:GOTO54
78 K=21-I*4:C=12-I*2:PRINT:FORI=0TOC:PRINTTAB((21-K)/2);LEFT$(M$(CH),K):NEXT:RET
URN
80 PRINT"VUOI LE ISTRUZIONI?"
82 GETA$:IFA$=""THEN82
84 IFA$="N"THEN110
86 POKE36879,27:PRINTCHR$(14);"*****STAI ATTENTO!*****"
88 PRINT"ORA TI MOSTRO LA PIAN-TA DI UN LABIRINTO"
90 PRINT"TU CI SEI DENTRO!":PRINT"LA TUA POSIZIONE E'INDICATA DALLA FRECCI
A"
92 PRINT"CHE MOSTRA ANCHE LA":PRINT"TUA DIREZIONE!":PRINT"    VEDRAI TUTTO I
N   "
94 PRINT"    3-DIMENSIONI!      ":PRINT"    PREMI UN TASTO"
96 GETR$:IFR$=""THEN96
98 PRINT"PER AIUTARTI LASCERAI":PRINT"LE IMPRONTE DOVE SEI":PRINT"GIA' PASSAT
O"
100 PRINTTAB(9)""
102 PRINT"=AVANZI DI 1 PASSO":PRINT"=TI GIRI A SIN."
104 PRINT"=TI GIRI A DESTRA":PRINT"=TI GIRI INDIETRO"
106 PRINT"=RINUNCI":PRINT"    PREMI UN TASTO";
108 GETR$:IFR$=""THEN108
110 PRINT"";CHR$(142):POKE36879,8:B=INT(RND(0)*20+1):FORI=0TO21:IFI=BTHENPOKEV+
I,32:GOTO114
112 POKEV+I,102
114 NEXT
116 FORR=1TO21:POKEV+R*22,102:POKEV+R*22+21,102:FORC=1TO20:A=INT(RND(1)+.5)
118 IFPEEK(V+R*22+C-22)=102THENPOKEV+R*22+C,102-70*A:GOTO124
120 IFPEEK(V+R*22+C-21)=32ANDPEEK(V+R*22+C-23)=32THENPOKEV+R*22+C,102:GOTO124
122 IFPEEK(V+R*22+C-23)=32ANDPEEK(V+R*22+C-1)=32THENPOKEV+R*22+C,102:GOTO124
123 POKEV+R*22+C,32
124 NEXT:NEXT:FORC=0TO21:POKEV+484+C,102:NEXT:FORI=0TO505:POKEW+I,PEEK(V+I):NEXT
:C=21-B
126 D=C+V+396
128 IFPEEK(D)=102THENC=C+1:GOTO126
130 PS=W+396+C:POKEPS,46:POKED,30
132 FORI=1TO30:POKED,188-PEEK(D):FORM=1TO300:NEXT:NEXT:PRINT"":RETURN
134 PRINT""
136 FORI=0TO505:POKEV+I,PEEK(W+I):NEXT:PRINT"":FORI=0TO20:PRINT"";:NEXT:PRINT"
VUOI GIOCARE?";
138 GETA$:IFA$=""THENPOKEPS+E,220-PEEK(PS+E):GOTO138
140 IFA$="S"THENPRINT"":GOTO6
142 IFA$="N"THENPRINT"CIAO!":POKE649,10:END
144 GOTO138
146 A$=TI$:PRINT"*****BRAVISSIMO!!*****"
148 PRINT"SEI USCITO IN:"
150 PRINTLEFT$(A$,2);"ORE ";MID$(A$,3,2);"MIN ";RIGHT$(A$,2);"SEC"
152 PRINT"ORA TI MOSTRO LA":PRINT"STRADA PERCORSA":FORI=0TO5000:NEXT:GOTO13
4
```

Il taglio del TI99/4A

Un avvincente gioco di strategia

di *Alessandro Chessa*

Non molto tempo fa, nel numero di Dicembre dell'anno scorso della rivista scientifica "Le Scienze", appariva, nella consueta rubrica "Temi Matematici" condotta brillantemente da Douglas R. Hofstadter, un interessante articolo intitolato "Taglio, Sfoggio, Hruska, evoluzione del comportamento e altri giochi di strategia", nel quale venivano presentati alcuni originali giochi numerici a carattere strategico-psicologico.

Tra i vari giochi, del resto tutti molto accattivanti, uno in particolare aveva stimolato la mia curiosità mentale. Intendo riferirmi al "Taglio", un gioco ideato dallo stesso autore e da un suo collega, Robert Boeninger. Le regole del gioco sono a dir poco semplici. Si tratta di scegliere mentalmente, ad ogni turno, due numeri, uno per ciascun giocatore (si gioca in due), "pescandoli" da uno stesso insieme numerico (numeri interi da 1 a 5) per poi confrontarli tra di loro; se la loro differenza risulta diversa da 1, ciascun giocatore terrà il proprio numero sommandolo al proprio punteggio; se invece la differenza è esattamente 1, il giocatore con il più basso dei numeri li sommerà entrambi, "tagliando" l'avversario che rimarrà all'asciutto. Quest'ultimo espediente è stato naturalmente escogitato per evitare che si possano scegliere troppo facilmente numeri grandi, rendendo così il gioco più movimentato e più articolato nel suo svolgimento.

Quello che più mi affascinava era la straordinaria possibilità, troppo spesso disattesa da sofisticati (come grafica) ma pur sempre limitati giochi spaziali, di prestarsi ad una infinità di soluzioni e di creare una partita che accumulasse livelli su livelli di gioco, limitati solo dalla fantasia e dalla disponibilità di tempo dei giocatori.

La bravura in questo gioco consiste nell'attivare una strategia fatta di astuzie psicologiche appositamente congeniate per sviare e poi "tagliare" l'avversario. Ciò si può ottenere per esempio, ostentando un determinato schema numerico, quale 4-4-4, per spingere l'avversario a dire 3 proprio nel momento in cui voi direte 2 (non sempre capita!).

È un gioco fatto di sottili tensioni psicologiche che spesso si risolvono in una serrata battaglia che ha come unico superstite (vincitore) il più saldo e forte di nervi.

Essendo possessore di un computer (TI99/4A) intravvedevo la possibilità, del resto già prospettata dallo stesso Hofstadter, di fare del mio computer un valente giocatore di Taglio.

Problemi e idee per un programma "esperto" di Taglio

Le maggiori difficoltà incontrate nell'ideare un programma "esperto" nel gioco del Taglio derivavano dal

fatto che non trattandosi di un gioco "deterministico" o quasi come ad esempio gli scacchi, non ci si poteva avvalere di tecniche euristiche, tipiche delle applicazioni nel campo dell'intelligenza artificiale. Per il taglio si deve più propriamente parlare di gioco "probabilistico", nel senso che la strategia di gioco si può fondare solo su previsioni, più o meno probabili, ricavate attraverso una ricerca statistica delle giocate precedenti che riveli delle ricorsività di schemi numerici, dove per schemi numerici intendo sequenze ben determinate di numeri. Occorreva, quindi, che il computer cogliesse gli schemi, tal volta anche involontari (e questo l'ho sperimentato personalmente), dell'avversario e, giocando in effetti sulla sua psicologia, se ne servisse per generare previsioni ragionevolmente fondate; ma questo non era evidentemente sufficiente. A questo riguardo vorrei fare un esempio abbastanza significativo. Se nel corso di una partita, per due giocate consecutive scegliamo il numero 1, il computer alla terza avendo individuato lo schema numerico 1-1, si aspetterà di nuovo un 1 e sarà quindi portato a giocare un 5, dato che in questa situazione è la scelta più vantaggiosa (5−1= +4). Naturalmente conoscendo già in precedenza la sua scelta, lo potremo facilmente tagliare con un 4, guadagnando ben 9 punti. Ora, pur ammettendo che in tutti e tre i turni il computer abbia giocato un 5, la somma totale dei punteggi risulta comunque a nostro favore. È chiaro che se provassimo a ripetere il tranello non incorrerebbe nello stesso errore, ma per evitare che lo stesso si possa verificare per altri schemi numerici, diversi da 1-1, si rende evidentemente necessaria una vera e propria strategia di gioco che diminuisca la prevedibilità delle mosse del computer.

Il mio programma cerca di realizzare tutto ciò sopra riferito.

Funzionamento del programma

Adesso in breve tenterò di spiegare come effettivamente funziona.

Una volta inserito il numero del giocatore (N) appare la scritta

```
1 REM *********************
2 REM *********************
3 REM ****** TAGLIO *******
4 REM *********************
5 REM *********************
6 REM *********************
7 REM * PERSONAL SOFTWARE *
8 REM *********************
9 REM *********************
10 REM
12 REM
14 REM
16 REM
20 REM
21 REM --------------------
22 REM   Gioco Strategico
24 REM   Ideato da:
26 REM Douglas R.Hofstadter
28 REM tratto da"Le Scienze"
30 REM --------------------
32 REM ********************
34 REM --------------------
36 REM Versione Computer di:
38 REM Alessandro Chessa
40 REM Su Home Computer:              TI 99/4A(Texas I. )
41 REM Versione Basic:
42 REM *EXTENDED BASIC*
44 REM
46 REM Cagliari 17/4/1983
47 REM --------------------
48 REM ********************
50 REM   Bibliografia:
52 REM   Le Scienze N 172
54 REM   Dicembre/82
56 REM ********************
58 REM
59 REM
60 REM ******************
61 REM *****VARIABILI*****
62 REM ******************
63 REM
64 REM X(200)... VETT. PRINC.
65 REM Y(200)... VETT. SECON.
66 REM N.....NUM. GIOCATORE
67 REM C... CONTATTORE VETT.
68 REM NC... NUM. COMPUTER
69 REM NA...NUM. AVVERSARIO
70 REM CAS1  *
71 REM CAS2,P**NUM. CASUALI
72 REM CAS3  *
73 REM C1*
74 REM C2**
75 REM C3***CONTATTORI
76 REM C4***SUBROUTINES
77 REM C5**
78 REM C6*
79 REM TOT.. PUNTEGG. GIOCAT.
80 REM CTOT. PUNTEGG. COMPUT.
81 REM W... C1,C2,C3,C4,C5
82 REM A$... VALIDATE:"SsNn"
83 REM DIF ...NC-N
84 REM T.....NUMERO TURNI
85 REM ******************
86 REM *****COSTANTI*****
87 REM ******************
89 REM Z... 1,2,3,4,5
90 REM ******************
91 REM **DIMENSIONAMENTO**
92 REM ******************
93 OPTION BASE 1
94 DIM X(200)
95 DIM Y(200)
96 REM
97 REM *****************
98 REM *INIZIALIZZAZIONE*
99 REM *****************
100 REM
105 LET C=0
110 LET NC=0
115 LET NA=0
120 LET C6=0
125 LET TOT=0
130 LET CTOT=0
132 REM
180 CALL CHAR(130,"3333CCCC3333CCCC")
182 CALL COLOR(13,6,8)
185 ON WARNING NEXT
190 RANDOMIZE
195 CALL SCREEN(10)
199 CALL CLEAR
200 PRINT "****************************"
203 PRINT "****************************"
205 PRINT "****************************"
210 PRINT "****************************"
213 PRINT "****************************"
215 PRINT "****************************"
220 PRINT "****************************"
225 PRINT "***********TAGLIO***********"
230 PRINT "****************************"
233 PRINT "****************************"
235 PRINT "****************************"
240 PRINT "****************************"
243 PRINT "****************************"
245 PRINT "****************************"
248 PRINT "****************************"
250 PRINT : : : :
251 CALL DELAY(2)
253 RESTORE 9500
255 READ A,B
257 IF A=-1 THEN 268
259 DISPLAY AT(A,B)SIZE(1):" "
261 CALL SOUND(-120,40000,30,40000,30,330,30,-8,1)
265 GOTO 255
268 CALL DELAY(3)
270 CALL SCREEN(8)
```

Listato 1. *Il programma BASIC.*

Seguito listato 1.

```
280 DISPLAY AT(10,7)ERASE ALL "VUOI CONOSCERE "
290 DISPLAY AT(12,3) "LE REGOLE DEL *TAGLIO*"
300 DISPLAY AT(15,9) "(S/N)?"
310 ACCEPT AT(15,16)BEEP SIZE(1)VALIDATE("SsNn") A$
320 IF A$="S" OR A$="s" THEN GOSUB 9000 :: CALL SCREEN(8)
322 DISPLAY AT(10,3)ERASE ALL "QUANTI TURNI VUOI GIOCARE"
324 DISPLAY AT(15,6) "(DA 50 A 200)?"
326 ACCEPT AT(15,22)BEEP VALIDATE(DIGIT)SIZE(-3) T
328 IF T<50 OR T>200 OR INT(T)<>T THEN 326
338 CALL SCREEN(8)
339 CALL CLEAR
340 DISPLAY AT(1,3) "PLAYER   TURN  COMPUTER"
345 IF T=C THEN 650
350 DISPLAY AT(2,3) "SCORE    ";C :: DISPLAY AT(2,18) "SCORE"
360 DISPLAY AT(3,4)SIZE(4) TOT :: DISPLAY AT(3,19)SIZE(4) CTOT
365 CALL HCHAR(5,1,130,32)
370 DISPLAY AT(9,5) "PLAYER       COMPUTER"
380 DISPLAY AT(10,5) "NUMBER       NUMBER"
385 DISPLAY AT(13,7)SIZE(14) "+            +"
387 CALL HCHAR(19,1,130,32)
390 ACCEPT AT(13,7)BEEP VALIDATE(DIGIT)SIZE(-1) N
395 IF N<1 OR N>5 OR INT(N)<>N THEN 385
400 DISPLAY AT(16,17)BEEP "COMPUTER" :: DISPLAY AT(17,17) "THINKING"
500 GOSUB 1000
510 LET C6=0
520 DISPLAY AT(13,19)BEEP NC
530 LET DIF=NC-N
540 IF DIF=1 THEN TOT=TOT+N+NC :: CALL SCREEN(14) :: DISPLAY AT(21,17) "TAGLIATO!
" :: CALL DELAY(3) :: DISPLAY AT(21,1) "" :: GOTO 580
550 IF DIF=-1 THEN CTOT=CTOT+NC+N :: CALL SCREEN(14) :: DISPLAY AT(21,5) "TAGLIAT
O!" :: CALL DELAY(3) :: DISPLAY AT(21,1) "" :: GOTO 580
560 LET TOT=TOT+N
570 LET CTOT=CTOT+NC
580 CALL SCREEN(8)
590 CALL DELAY(3)
600 CALL HCHAR(16,1,32,64):: GOTO 345
650 REM ******************
655 REM **MESSAGGI FINALI**
660 REM ******************
663 CALL SCREEN(12)
665 DISPLAY AT(2,7)BEEP ERASE ALL "PUNTEGGIO FINALE"
670 DISPLAY AT(5,4):"PLAYER         COMPUTER"
673 DISPLAY AT(6,5) TOT :: DISPLAY AT(6,19):CTOT
675 IF CTOT>TOT THEN DISPLAY AT(15,4):"  EBBENESI'   SONO ANCORA IO   ER/MEJO DE'
TUTTI."
680 IF CTOT=TOT THEN DISPLAY AT(15,9):"    PER QUESTA VOLTA TI E'      ANDATA BE
NE                 E' FINITA IN PARITA'."
685 IF CTOT<TOT THEN DISPLAY AT(15,4):"   BRAVO!!! SEI RIUSCITO A     BATTERE UN
 COMPUTER."
690 CALL DELAY(20)
700 DISPLAY AT(8,5)ERASE ALL "VOUI INIZIARE UNA"
710 DISPLAY AT(10,5):"NUOVA PARTITA"
720 DISPLAY AT(15,7)SIZE(6):"(S/N)?"
730 ACCEPT AT(15,16)BEEP SIZE(1)VALIDATE("SNsn"):A$
750 IF A$="S" OR A$="s" THEN 90
760 DISPLAY AT(16,9)ERASE ALL "ARRIVEDERCI!!"
850 STOP
900 REM ******************
902 REM ******************
903 REM ****SUBROUTINES****
905 REM ******************
906 REM ******************
907 REM
909 REM
910 REM
950 REM ******************
960 REM   SUBROUTINE 1000
970 REM (RICERCA STRATEGIA)
980 REM ******************
990 REM
999 REM
1000 LET C=C+1
1010 LET X(C)=N
1013 LET CAS1=INT(RND*5)+1
1015 LET CAS2=INT(RND*5)+1
1017 IF C>9 THEN 1070
1018 REM
1019 REM ******************
1020 REM STRATEGIA INIZIALE
1022 REM ******************
1023 REM
1024 IF C=1 THEN NC=CAS2 :: RETURN
1026 IF C=2 THEN IF CAS1=1 AND X(C-1)<>1 THEN NC=X(C-1)-1 :: RETURN ELSE NC=CAS2
 :: RETURN
1030 IF C=3 THEN IF X(C-1)=X(C-2)AND CAS1>1 THEN NA=X(C-1) :: GOTO 1080 ELSE NC=C
AS2 :: RETURN
1050 ON C-3 GOSUB 5500,5000,5000,5000,5000,5000
1060 GOTO 1080
1070 GOSUB 2000
1073 REM ******************
1075 REM     STRATEGIA
1077 REM ******************
1080 LET P=RND
1100 IF NA=0 AND T-C<=13 THEN IF CTOT-TOT<=-20 THEN NC=INT(RND*3)+3 :: RETURN
1110 IF NA=0 AND T-C<=13 THEN IF CTOT-TOT>=20 THEN NC=INT(RND*3)+1 :: RETURN
1120 IF NA=0 THEN NC=INT(RND*5)+1 :: RETURN
1140 IF NA=1 AND CTOT-TOT<=-20 THEN IF P<.70 THEN NC=5 :: RETURN ELSE NC=4 :: RE
TURN
1150 IF NA=1 AND CTOT-TOT>=20 AND P<.75 THEN IF P<.40 THEN NC=5 :: RETURN ELSE N
C=4 :: RETURN
1160 IF NA=1 AND CTOT-TOT>=20 THEN NC=3 :: RETURN
1170 IF NA=1 THEN IF P<.60 THEN NC=5 :: RETURN ELSE NC=4 :: RETURN
1180 IF NA=2 AND CTOT-TOT<=-20 THEN IF P<.70 THEN NC=1 :: RETURN ELSE NC=5 :: RE
TURN
```

"Computer Thinking" e il computer inizia la sua ricerca ignorando naturalmente il numero appena inserito. Dapprima considera gli ultimi 5 numeri scelti e va a ricercare se precedentemente sia stato già giocato quello schema. Ogni qualvolta ne incontra uno identico inserisce in un contenitore (il vettore Y) il numero subito seguente lo schema. Se alla fine della ricerca non ha trovato nessuno schema da 5 numeri passa a considerare schemi da 4 poi da 3, da 2 ed infine da 1 numero. Ammettiamo, per semplicità, che abbia individuato 10 schemi da 3 numeri e che i numeri seguenti questi schemi, ovvero i numeri di preferenza in quelle particolari circostanze, siano stati due 2 e otto 5. A questo punto il computer sceglie dal contenitore un numero (NA = probabile numero avversario) fra i dieci prescelti. Bisogna però considerare che la casualità è fortemente condizionata dalle proporzioni in cui sono stati giocati gli schemi. In questo caso particolare la scelta è ristretta ai numeri 2 e 5 e la probabilità che esca il 5 è di 8 parti su 10 (80% contro il 20% del 2). Una volta estratto il numero dal contenitore si procede all'applicazione della strategia. La strategia da me ideata fa uso di determinati *pesi*, espressi in percentuali, che sono assegnati alle diverse scelte possibili (tabella 2). Le diverse scelte sono a loro volta determinate dal numero estratto dal contenitore. I criteri adottati per la "calibrazione" dei pesi sono del tutto soggettivi, e sono quindi passibili di modifiche a seconda delle diverse interpretazioni tattiche. In generale, per quanto riguarda il criterio da me seguito, posso dire di aver tenuto conto di tre fattori fondamentali: il guadagno nel punteggio, la "tagliabilità" del numero e le condizioni di gioco contingenti. Mi spiegherò meglio con un esempio. Se il numero previsto dal computer (NA) è un 2 sarebbe indifferente per lui giocare un 1 od un 5 perché entrambi procurerebbero un più 3 nel conteggio del turno (infatti 1 + 2 = 5 − 2). Non sempre però il numero previsto è quello effettivamente scelto dall'avversario. Nell'incertezza l'1 è da preferirsi in quanto non può essere tagliato

Seguito listato 1.

```
1190 IF NA=2 AND CTOT-TOT>=20 AND P<.80 THEN IF P<.50 THEN NC=1 :: RETURN ELSE N
C=5 :: RETURN
1200 IF NA=2 AND CTOT-TOT>=20 THEN NC=4 :: RETURN
1210 IF NA=2 THEN IF P<.65 THEN NC=1 :: RETURN ELSE NC=5 :: RETURN
1220 IF NA=3 AND CTOT-TOT<=-20 THEN IF P<.80 THEN NC=2 :: RETURN ELSE NC=5 :: RE
TURN
1230 IF NA=3 AND CTOT-TOT>=20 AND P<.85 THEN IF P<.55 THEN NC=2 :: RETURN ELSE N
C=5 :: RETURN
1240 IF NA=3 AND CTOT-TOT>=20 THEN NC=3 :: RETURN
1250 IF NA=3 THEN IF P<.75 THEN NC=2 :: RETURN ELSE NC=5 :: RETURN
1260 IF NA=4 AND CTOT-TOT<=-20 THEN IF P<.90 THEN NC=3 :: RETURN ELSE NC=4 :: RE
TURN
1270 IF NA=4 AND CTOT-TOT>=20 THEN IF P<.80 THEN NC=3 :: RETURN ELSE NC=4 :: RET
URN
1280 IF NA=4 THEN IF P<.85 THEN NC=3 :: RETURN ELSE NC=4 :: RETURN
1290 IF NA=5 AND CTOT-TOT<=-20 THEN IF P<.95 THEN NC=4 :: RETURN ELSE NC=5 :: RE
TURN
1300 IF NA=5 AND CTOT-TOT>=20 THEN IF P<.85 THEN NC=4 :: RETURN ELSE NC=5 :: RET
URN
1310 IF NA=5 THEN IF P<.90 THEN NC=4 :: RETURN ELSE NC=5 :: RETURN
1898 REM
1900 REM ******************
1905 REM *   SUBROUTINES  *
1910 REM *     RICERCA    *
1915 REM *     SCHEMI     *
1920 REM ******************
1925 REM
1930 REM
1935 REM
1940 REM ******************
1945 REM  SUBROUTINE 2000
1950 REM  SCHEMI A 5 NUM.
1955 REM ******************
2000 FOR C1=1 TO C-8 !(C-1-3-4)
2010 LET W=C1
2020 LET Z=5
2030 IF X(C-1)=X(C1+4)AND X(C-2)=X(C1+3)AND X(C-3)=X(C1+2)AND X(C-4)=X(C1+1)AND
X(C-5)=X(C1)THEN GOSUB 6000
2040 NEXT C1
2050 IF C6<>0 THEN GOSUB 7000 ELSE GOSUB 3000
2070 RETURN
2900 REM
2905 REM
2910 REM
2915 REM ******************
2920 REM  SUBROUTINE 3000
2925 REM  SCHEMI A 4 NUM.
2930 REM ******************
3000 FOR C2=1 TO C-7 !(C-1-3-3)
3010 LET W=C2
3020 LET Z=4
3030 IF X(C-1)=X(C2+3)AND X(C-2)=X(C2+2)AND X(C-3)=X(C2+1)AND X(C-4)=X(C2)THEN G
OSUB 6000
3040 NEXT C2
3050 IF C6<>0 THEN GOSUB 7000 ELSE GOSUB 4000
3060 RETURN
3900 REM
3905 REM
3910 REM
3915 REM ******************
3920 REM  SUBROUTINE 4000
3925 REM  SCHEMI A 3 NUM.
3930 REM ******************
4000 FOR C3=1 TO C-5 !(C-1-2-2)
4010 LET W=C3
4020 LET Z=3
4030 IF X(C-1)=X(C3+2)AND X(C-2)=X(C3+1)AND X(C-3)=X(C3)THEN GOSUB 6000
4040 NEXT C3
4050 IF C6<>0 THEN GOSUB 7000 ELSE GOSUB 5000
4060 RETURN
4900 REM
4905 REM
4910 REM
4915 REM ******************
4920 REM  SUBROUTINE 5000
4925 REM  SCHEMI A 2 NUM.
4930 REM ******************
5000 FOR C4=1 TO C-3 !(C-1-1-1)
5010 LET W=C4
5020 LET Z=2
5030 IF X(C-1)=X(C4+1)AND X(C-2)=X(C4)THEN GOSUB 6000
5040 NEXT C4
5050 IF C6<>0 THEN GOSUB 7000 ELSE GOSUB 5500
5060 RETURN
5400 REM
5410 REM
5420 REM
5430 REM
5440 REM ******************
5450 REM  SUBROUTINE 5500
5460 REM  SCHEMI A 1 NUM.
5470 REM ******************
5500 FOR C5=1 TO C-2 !(C-1-1)
5510 LET W=C5
5520 LET Z=1
5530 IF X(C-1)=X(C5)THEN GOSUB 6000
5540 NEXT C5
5550 IF C6<>0 THEN GOSUB 7000 ELSE LET NA=0
5560 RETURN
5900 REM
5905 REM
5910 REM
5915 REM
```

da nessun numero. C'è da aggiungere ancora che se veramente il numero previsto non è giusto, in questo caso è diverso da 2, l'1 non guadagna più +3 punti, perché non sussiste più il taglio. L'ultima considerazione da fare è relativa alle condizioni di gioco contingenti. Nel caso sia in vincita (+ 20) il computer allargherà le sue possibili scelte, anche a 3 numeri, per rendere meno prevedibili le sue mosse e amministrare "saggiamente" il suo vantaggio. Nel caso sia in perdita (− 20) opterà per una tattica il più fruttuosa possibile.

Come si vede le considerazioni sono molteplici e tutte quante insieme concorrono alla calibrazione dei pesi adoperati. Vi è infine l'ultima possibilità che la ricerca degli schemi non abbia avuto i frutti sperati (questo capita sempre nei primi turni di gioco). In questo caso si procederà alla generazione di numeri casuali che saranno condizionati dalle condizioni di gioco contingenti.

Il programma passo, passo...

1-89 *Righe di commento.* Qualcuno obietterà che abbia ecceduto nei "commenti", ma, come giustamente predica **Personal Software** (N. 3, pag. 23, regola 1), un buon programmatore deve sapersi autodocumentare.

90-199 *Qui si esplicano le consuete procedure di dimensionamento e inizializzazione.* Da notare L'OPTION BASE 1 alla riga 93 che serve per inizializzare ad 1 l'indice di partenza dei vettori, che nel TI Extended BASIC è di norma 0. C'è da osservare inoltre alla riga 185, l'istruzione ON WARNING NEXT, altra specifica del TI Extended BASIC, che permette di controllare la segnalazione di WARNING in caso di errore nel dato di input, cosa che scombussolerebbe tutto il quadro di gioco. Il NEXT fa sì che venga ignorato l'errore.

200-850 *Visualizzazione del quadro di gioco.* Dapprima si richiede al giocatore se vuole conoscere le regole del gioco e in quanti turni desidera giocare (da 50 a 200 turni). Poi, dopo aver disegnato il quadro di gioco, attende il numero dell'av-

Seguito listato 1.

```
5920 REM ******************
5925 REM  SUBROUTINE 6000
5930 REM CARICAMENTO[Y(C6)]
5935 REM ******************
6000 LET C6=C6+1
6010 LET Y(C6)=X(W+Z)
6020 RETURN
6900 REM
6905 REM
6910 REM
6915 REM ******************
6920 REM  SUBROUTINE 7000
6925 REM  SCELTA DI -NA-
6930 REM ******************
7000 LET CAS3=INT(RND*C6)+1
7010 LET NA=Y(CAS3)
7020 RETURN
7030 REM
7040 REM
8900 REM ******************
8910 REM ******REGOLE******
8920 REM ******************
9000 CALL SCREEN(11)
9010 CALL CLEAR
9020 PRINT "   Le regole del Taglio:"
9030 PRINT
9040 PRINT "Questo e` un gioco strategi co, e consiste nel trvare la giusta stra
tegia psicologi ca per confondere e sopraffa re l'avversario(Computer) "
9050 PRINT "Si gioca in 50<>200 turni. A ogni turno dovete scegliere un numero(In
tero da 1 a 5) e inserirlo nel computer; il"
9060 PRINT "computer a sua volta ne pen sera` uno A questo punto si  confrontera
nno i due numeri"
9070 PRINT "Se la loro differenza sara` diversa da 1, allora ognuno sommera` il
suo numero al proprio punteggio, se invece"
9080 PRINT "la differenza sara`esattamente 1, allora il giocatore   con il piu`
basso dei numeri li sommera` entrambi,mentre"
9090 PRINT "l'altro verra` -TAGLIATO- "
9099 CALL KEY(0,K,S)
9100 IF S<=0 THEN 9099 ELSE CALL CLEAR :: RETURN
9500 DATA 5,28,6,27,6,26,7,25,8,25,8,,24,9,23,10,22,10,21,11,20,12,20,13,19,13,18
,14,17
9510 DATA 14,16,15,15,15,14,14,13,13,12,12,11,13,10,14,10,15,9,16,8,17,7,17,6,18
,5,19,4,-1,-1
9900 REM ******************
9910 REM  SUBROUTINE DELAY
9920 REM ******************
10000 SUB DELAY(SEC)
10010 FOR COUNT=1 TO 200*SEC
10020 NEXT COUNT
10030 SUBEND
20000 END
```



versario (N) e, solo ad inserimento avvenuto, provvederà a sua volta a "pensarne" uno (appare la scritta "COMPUTER THINKING") e visualizzarlo nella giusta posizione (righe 200-520). Si procede poi al conteggio e alla eventuale segnalazione di un taglio (righe 530-600). Alla fine dei turni stabiliti in precedenza, compariranno il punteggio finale e i soliti messaggi scherzosi di commiato. Inoltre si darà l'opportunità di giocare una nuova partita (righe 650-850).

Dalla riga 500 si accede alle subroutine per la ricerca degli schemi. La riga 510 provvede all'opportuno azzeramento del contatore della subroutine 6000. Le istruzioni ACCEPT-AT consentono di accettare dati da qualsiasi parte dello schermo e inoltre, cosa molto importante, controllano i dati in entrata, rifiutandoli se non sono conformi alle caratteristiche specificate nella VALIDATE. Le CALL DELAY si riferiscono al sottoprogramma DELAY (righe 10000-10030) e provocano delle pause lunghe tanti secondi, quanti specificati nell'argomento tra parentesi.

Dalla riga 900 cominciano le subroutine.

1000 *Ricerca strategica.* Prima di tutto viene incrementato il contatore dei turni (C) e viene inserito nel vettore X il numero scelto dal giocatore (N); poi sono generati due numeri casuali (CAS1 e CAS2) e, se sono stati giocati più di 9 turni, si salta la "strategia iniziale", riservata appunto ai primi 9 turni che sono casisticamente più facili da controllare. Dalla riga 1080 si effettua la ricerca strategica vera e propria. Per ottenere i pesi di cui sopra riferito ho provveduto alla generazione di numeri casuali. Per fare in modo, per esempio, che l'1 avesse il 65% di probabilità di essere scelto nei confronti del 5 (35%) è stata sufficiente la seguente condizione logica:

```
IF P < .65 THEN NC = 1 ELSE
NC = 5
dove P = RND.
```

Quando invece si è trattato di "pesare" tre numeri allora si è dovuto sdoppiare l'istruzione per non creare una riga troppo complessa.

2000, 3000, 4000, 5000, 5500 *Qui si procede alla ricerca degli schemi numerici rispettivamente da 5, 4, 3, 2 e 1 numero.*

Ogni subroutine consta di un loop e di una condizione logica finale. All'interno del loop vi sono tre istruzioni; la terza è una istruzione logica che controlla numero per numero se lo schema è verificato; in tal caso richiama la subroutine 6000 e individuato con le prime due istruzioni il numero seguente lò schema lo si inserisce nel vettore Y. All'uscita del loop, quando tutti gli schemi sono stati provati, viene verificato se il vettore Y sia stato riempito. Nel caso sia vuoto si passa alla subroutine ricerca schemi successiva, altrimenti viene richiamata la subroutine 7000.

6000, 7000 *Accumulazione e scelta di NA.* La subroutine 6000 viene richiamata ogni qualvolta è stato verificato uno schema ed è quindi necessario conservare il valore di NA nel vettore Y. La subroutine 7000, una volta terminata la ricerca degli schemi, estrae casualmente dal vettore Y un valore di NA.

È importante ricordare che per far girare questo programma è necessario il modulo SSS TI Extended BASIC senza il quale sarebbe stato forse impossibile redigere un programma di questo genere, non possedendo il TI BASIC normale la fondamentale istruzione logica IF-THEN-ELSE estesa.

Conclusioni

È stato entusiasmante accorgersi come il computer, dopo un avvio non troppo brillante, iniziava a "fiutare" i primi schemi numerici e ad operare i primi tagli decisivi. È inevitabile che ciò accada perché prima o poi più o meno tutti siamo portati inconsciamente a schematizzare le nostre azioni secondo una logica che non sempre siamo in grado di rilevare a livello conscio. Il computer fa pressochè totale affidamento proprio su questa debolezza umana.

Non è comunque questo uno di quei casi in cui si può parlare del computer come di un giocatore imbattibile, ma vi posso assicurare che è un avversario tenace, soprattutto se vi mettete in testa di tendergli qualche tranello. Con lui potrete intrattenere delle interessanti partite.

In definitiva mi sembra di poter dire di avere ottenuto un programma capace di giocare, per lo meno alla pari, con il suo avversario. Può senz'altro essere migliorato e soprattutto modificato in molte parti con particolare riferimento alla strategia di gioco che, a mio avviso, può essere "interpretata" in diversi modi. Invito quindi i lettori a dare la loro interpretazione e a sfidare, in una battaglia computer contro computer, il mio programma. ■

VARIABILI	DESCRIZIONE
X (Max. 200)	*Vettore principale (conserva i valori di N)*
Y (Max. 200)	*Vettore secondario (conserva i valori di NA)*
C	*Contatore vettoriale (X)*
NC	*Numero del computer*
NA	*Numero dell'avversario (probabile)*
CAS1, CAS2, CAS3	*Numeri casuali*
P (RD)	*Numero casuale per generare i pesi*
C1, C2, C3, C4, C5, C6	*Contatori delle subroutine*
TOT	*Totale del giocatore*
CTOT	*Totale del computer*
T	*Numero dei turni*
W, Z, A$	*Variabili di lavoro*
N	*Numero del giocatore*

Tabella 1. *Variabili del programma.*

PROBABILE NUMERO AVVERSARIO	NUMERO COMPUTER
	Normali
1	5 (60%) – 4 (40%)
2	1 (65%) – 5 (35%)
3	2 (75%) – 5 (25%)
4	3 (85%) – 4 (15%)
5	4 (90%) – 5 (10%)
	In perdita (–20)
1	5 (70%) – 4 (30%)
2	1 (70%) – 5 (30%)
3	2 (80%) – 5 (20%)
4	3 (90%) – 4 (10%)
5	4 (95%) – 5 (5%)
	In vincita (+20)
1	5 (40%) – 4 (35%) – 3 (25%)
2	1 (50%) – 5 (30%) – 4 (20%)
3	2 (55%) – 5 (30%) – 3 (15%)
4	3 (80%) – 4 (20%)
5	4 (85%) – 5 (15%)

Tabella 2. *Pesi per la strategia di gioco, espressi in percentuali. Il programma sceglie il proprio numero (NC) in funzione delle condizioni di gioco (NORMALI, IN PERDITA, IN VINCITA) e sulla base del probabile numero dell'avversario (NA).*

COMMODORE VIC 20

RESTORE nn per il VIC 20

di Alessandro Guida

Il VIC 20 si è sempre dimostrato un piccolo prodigio, pur risentendo della mancanza di alcune istruzioni BASIC tradizionali. Abbiamo già visto in **Personal Software** n. 7 come ottenere il PRINT AT. Questa volta implementeremo il comando RESTORE nn. Sarà un'ottima occasione per approfondire la conoscenza del sistema operativo.

Sicuramente conoscerete bene le istruzioni DATA, READ e RESTORE. In particolare il RESTORE riposiziona il puntatore, che indica il prossimo dato da leggere, all'inizio del primo DATA. Ciò è sicuramente limitativo, infatti spesso non si ha nessun interesse a rileggere tutti i dati dall'inizio. Un esempio è l'aver ordinato alcune informazioni come nel listato 2. In questo caso sarebbe utile poter cominciare a leggere con l'istruzione READ dall'inizio di una linea qualsiasi (purchè contenente una frase DATA). È proprio questa l'utilità del RESTORE nn, dove con nn indichiamo un numero di riga qualsiasi.

La routine che aggiunge al VIC, in qualsiasi configurazione, questo comando è riportata nel listato 1. L'uso è molto semplice: basta chiamare una volta la subroutine che dopo può anche essere cancellata. A questo punto quando vogliamo servirci del RESTORE nn sarà sufficiente l'istruzione X=USR(nn). L'unica accortezza da usare nello scrivere programmi che incorporano questa routine è di non scrivere mai con delle POKE nelle locazioni $00, $01 e $02, perché queste sono utilizzate dalla istruzione USR().

Va notato, inoltre, che se il numero di linea nn impostato non esiste viene considerata la prima linea seguente, e che se la linea nn non contiene DATA si ha il messaggio "OUT OF DATA ERROR IN xx", con xx uguale al numero della linea che conteneva il X=USR(nn). La variabile X può avere qualsiasi altro nome.

La routine è costituita da poche righe in BASIC che servono a introdurre nei byte $00, $01, $02 i valori esatti e a caricare in memoria la subroutine in linguaggio macchina che viene chiamata ogni volta si voglia ottenere il RESTORE nn. La parte in linguaggio macchina è spiegata chiaramente in figura 2.

Come anticipato il listato 2 è un programmino esempio (utile per fare ripetere la geografia ai vostri figli), in cui è usata questa routine.

```
. :0400 20 F7 D7 20 13
. :0405 C6 A0 04 B1 5F
. :040A F0 07 C9 83 F0
. :040F 09 C8 D0 F5 A2
. :0414 0D 20 3A C4 EA
. :0419 18 98 65 5F 85
. :041E 41 A9 00 65 60
. :0423 85 42 60 EA EA
```

Figura 1. *Dump della routine descritta.*

```
.,0400 JSR $D7F7     'Converte floating point in intero
.,0403 JSR $C613     'Ricerca il numero di linea Basic contenuto in $14,$15.
.,0406 LDY #$04      '---------------------------------------------------
.,0408 LDA ($5F),Y   '
.,040A BEQ $0413     'Cerca l'istruzione DATA ($83) nella linea in esame
.,040C CMP #$83      '
.,040E BEQ $0419     '   se la trova salta a $0419
.,0410 INY           '
.,0411 BNE $0408     '---------------------------------------------------
.,0413 LDX #$0D      'Se non ha trovato l'istruzione DATA salta alla routine di
.,0415 JSR $C43A     '   trattamento degli errori con il codice $0D(OUT OF DATA)
.,0418 NOP           '---------------------------------------------------
.,0419 CLC           '
.,041A TYA           '
.,041B ADC $5F       'Aggiorna i puntatori DATA ($41,$42)
.,041D STA $41       '
.,041F LDA #$00      '
.,0421 ADC $60       '
.,0423 STA $42       '
.,0425 RTS           '---------------------------------------------------
```

Figura 2. *Routine in linguaggio macchina disassemblata e commentata.*

```
1000 HB=PEEK(56)-1:LB=PEEK(55):POKE0,76:POKE1,LB:POKE2,HB
1010 POKE56,HB:POKE52,HB:A=HB*256+LB
1020 RESTORE
1030 READN:IFN=-1THENEND
1040 POKEA,N:A=A+1:GOTO1030
1050 REM **************************************
1060 REM * L'ISTRUZIONE :     'RESTORE NN'      *
1070 REM * SI OTTIENE CON IL COMANDO A=USR(NN) *
1080 REM * SE LA LINEA NON CONTIENE FRASI DATA *
1090 REM * SI HA L'ERRORE 'OUT OF DATA ERROR'  *
1100 REM * SE INVECE IL NUMERO DI LINEA CITATO *
1110 REM * NON ESISTE ALLORA VIENE CONSIDERATA *
1120 REM * LA PRIMA LINEA SEGUENTE             *
1130 REM **************************************
1140 DATA032,247,215,032,019,198,160,004,177,095
1150 DATA240,007,201,131,240,009,200,208,245,162
1160 DATA013,032,058,196,234,024,152,101,095,133
1170 DATA065,169,000,101,096,133,066,096,234,234,-1
```

Listato 1. *Questa routine aggiunge al VIC 20 l'istruzione RESTORE nn.*

BASIC e linguaggio macchina

Il BASIC del VIC 20 dispone di due istruzioni per collegarsi con parti di programma scritte in linguaggio macchina: SYS(indirizzo) e USR(xx).

La prima è senz'altro la più conosciuta, e per questo la tralasciamo. La seconda, invece, è raramente utilizzata, pur avendo l'innegabile vantaggio di poter trasmettere direttamente dati numerici dal BASIC al linguaggio macchina e viceversa. La forma completa di questa istruzione è A=USR(xx). "xx" è il valore da trasmettere alla routine in linguaggio macchina, e può essere un numero o il nome di una variabile numerica. Al ritorno al BASIC la variabile A (può avere un nome qualsiasi) contiene il risultato dell'elaborazione. La prima cosa importante da ricordare è che quando l'interprete BASIC incontra l'istruzione USR avvia l'esecuzione della parte in linguaggio macchina partendo sempre dalla locazione $00. Per questo motivo i primi tre byte di memoria devono contenere sempre:

$00	contiene: $4C	(dec. 76) che corrisponde all'istruzione JMP di salto.
$01	contiene: $LB	byte meno significativo dell'indirizzo di inizio della routine in linguaggio macchina.
$02	contiene: $HB	byte più significativo.

Va, anche, ricordato che all'atto del passaggio al linguaggio macchina il numero xx viene conservato nel registro FAC1 (memoria da $61 a $66) e che ritornando al BASIC viene attribuito alla variabile A il valore FAC1 in quel momento.

Il FAC1 (*Floating Accumulator*) è una zona di memoria utilizzata dall'interprete BASIC per tutti i calcoli. In esso il numero è conservato in forma floating point (virgola mobile) nella seguente maniera:

$61 = esponente
$62 = MSB mantissa
$63 = mantissa
$64 = mantissa
$65 = LSB mantissa
$66 = segno ($FF = negativo; $00 = positivo).

Obiettivamente utilizzare numeri in questo formato non è né immediato né semplice. Possiamo, però, ricorrere a delle routine del sistema operativo che consentono il trattamento dei numeri in floating point, e che sono visibili in tabella 1.

Indirizzo	Descrizione
$D391	Converte un numero intero in floating point. Preleva il numero intero dall'accumulatore (byte alto) e dal registro Y (byte basso) e deposita il risultato nel FAC1.
$D7F7	Converte un numero in floating point in intero. Lo preleva dal FAC1 e deposita l'intero nei byte $14 e $15 (byte alto e basso).
$DDDD	Converte il numero in floating point nel FAC1 in una stringa pronta per essere stampata. La stringa è memorizzata a partire dalla locazione $0100 e termina con un byte uguale 0.

Tabella 1. *Routine del sistema operativo utilizzate per la conversione dei numeri in floating point.*

```
10 DIMA$(19)
20 GOSUB1000:REM ATTIVA RESTORE N
25 PRINT"■";
30 FORI=0TO19:A=USR(2000+I*10):REM RESTORE(NN)
40 READA$:PRINTITAB(5)LEFT$(A$,15)
50 NEXT
60 INPUT"■REGIONE";R$
70 IFR$<"0"ORVAL(R$)>20ORR$>"9"THENPRINT"■■";:GOTO60
80 PRINT"■":A=USR(2000+VAL(R$)*10):REM RESTORE
85 N=0
90 READA$(N):IFA$(N)<>"*"THENN=N+1:GOTO90
100 FORI=1TON-1:INPUT"PROV.";R$:PRINT"■";
110 R=0:FORH=1TON-1
120 IFR$=A$(H)THENA$(H)="*":H=N:R=1
130 NEXTH
140 IFR=0THEN300
150 PRINT"■PROV.? "R$"■"
160 NEXTI
170 PRINT"■■■BRAVO!!":PRINT"■■SCEGLI   UN'ALTRA    REGIONE"
180 FORI=0TO3000:NEXT:GOTO25
300 PRINT"■HAI SBAGLIATO!!"
310 PRINT"■■DOVRAI RIPETERE MEGLIO■LA REGIONE "A$(0):GOTO180
1000 REM ROUTINE LISTATO 1
1010 REM ATTIVAZIONE RESTORE N
2000 DATAVAL D'AOSTA,AOSTA,*
2010 DATALIGURIA,GENOVA,IMPERIA,SAVONA,LA SPEZIA,*
2020 DATACALABRIA,CATANZARO,COSENZA,REGGIO CALABRIA,*
2030 REM AGGIUNGERE I DATA DELLE ALTRE REGIONI
2040 REM DATA 'REGIONE','PROVINCIE','*'

READY.
```

Listato 2. *Esempio di utilizzo della nuova istruzione RESTORE nn.*

ERRATA CORRIGE

Nel numero 8-9 abbiamo presentato l'articolo "Gestione del cursore del VIC 20" a firma di Enrico Bossaglia anziché di Alessandro Guida. Ce ne scusiamo con gli interessati.

COMMODORE PET/CBM

Un overlay più flessibile

di Attilio Zannoni

Nella rubrica del n. 7 sono stati spiegati esaurientemente due tecniche utili per l'effettuazione dell'overlay tra programmi nel PET Commodore.

Ritengo che quello consigliato sia un metodo utile quando due programmi si devono scambiare valori di variabili, ma ci sono numerosi casi in cui un tale scambio non è necessario o è possibile effettuarlo in altro modo, per esempio creando un file di dati temporaneo su disco.

Si suggeriva di includere dei REM nei programmi per renderli di ugual lunghezza, ottenendo però un inutile spreco della memoria sia centrale che di massa. Anche con il secondo metodo consigliato (POKE opportuna nelle locazioni 43, 35, 47) si ha spreco, in questo caso solo di memoria quella centrale, infatti per il gioco dei puntatori, l'area di memoria esistente tra la fine del programma e l'inizio della memoria dati non verrà utilizzata perché noi abbiamo forzato i puntatori all'inizio memoria dati al valore necessario al programma di maggiori dimensioni.

Un metodo più comodo, che rende possibile collegare tra loro programmi di dimensioni diverse anche in tempi successivi, senza doverli modificare, prevede l'uso di due locazioni di memoria, uguali per le serie 3000, 4000 e 8000, che contengono il puntatore alla locazione di memoria successiva a quella di fine programma BASIC. Tali locazioni sono la 201 e la 202, che contengono rispettivamente la parte bassa e quella alta di tale indirizzo.

Basterà inserire come prima istruzione ai programmi che si richiamano in overlay:

```
10 POKE 43, PEEK (202): POKE 42, PEEK (201):
   CLR
```

Utilizzando questo sistema, se si interrompe l'esecuzione di un programma e lo si modifica, le locazioni 201 e 202 non vengono variate. Se lo si fa ripartire per controllare la modifica, l'istruzione precedentemente introdotta come prima, causerà il "taglio" della parte finale del programma, in misura proporzionale alla lunghezza di un'eventuale aggiunta, eliminando anche i caratteri di controllo di fine programma. Per evitare dei malfunzionamenti dovremo inserire la riga 10 solo quando i programmi sono corretti e, in caso di correzioni successive, richiamare direttamente il programma originale ed eventualmente eseguirlo facendo saltare la linea 10.

SINCLAIR ZX81 - SPECTRUM

Risparmiamo memoria

di Bruno del Medico

Queste note hanno come riferimento l'articolo Introduzione alla intelligenza artificiale che potete leggere in questo numero.

Utilizzo della linea 23 nello Spectrum

Nello ZX81 l'indirizzo 16418 contiene il numero di linee riservate al computer in basso nello schermo, generalmente 2, permettendo di utilizzare solo 22 delle 24 linee nominali. Volendo utilizzare anche le ultime due linee, si deve usare l'istruzione:

```
POKE 16418, 0
```

Per quello che concerne lo Spectrum, l'istruzione equivalente da usare per scrivere qualcosa nella linea 23 è:

```
PRINT # 1; AT 1, 0; "quello che vogliamo scrivere"
```

Nel momento in cui, viene eseguito un input la scritta scompare, il sistema migliore per mantenerla sul video è di usare: PAUSE.

Se si vuole cancellare la scritta prima di un eventuale input si scrive dopo il PAUSE:

```
PRINT # 1; AT 1, 0; "tanti spazi quanti caratteri scritti"
```

Se si vuole scrivere nella linea 22 si usa:

```
PRINT # 1; AT 0, 0; "XXXX"
```

è possibile ottenere uno scroll verso l'alto di ciò che è stampato alle linee 22 e 23 con il comando

```
PRINT # 1; AT 2, 0; "XXXX"
```

Il seguente programma evidenzia tale modo di operare:

```
10 FOR K = 0 TO 21
20 PRINT "LINEA", K
30 NEXT K
35 FORK = 0 TO 21
40 PRINT # 1; AT K, 0; "O.K."
45 PAUSE 20
50 NEXT K
```

Si può ottenere uno scroll immediato nella parte bassa con il comando:

```
PRINT # 1; AT N + 2, 0; " "
```

Sblocco dell'EDIT

Con il sistema da 1 Kbyte, succede spesso di caricare programmi che riempiono quasi al limite la memoria. Nel qual caso l'EDIT non funziona, vale a dire che la linea non scende in basso ed occorre riscriverla per intero; a volte non si riesce neppure a riscriverla. L'inconveniente si può ovviare battendo CLEAR e NEWLINE: il computer si sblocca e l'EDIT riprende a funzionare.

Risparmio di memoria con lo ZX81 (e Spectrum)

Non è immediatamente ovvio che la linea:

```
10 LET A = 25 + 37
```

occupa molta più memoria di:

```
10 LET A = VAL "25" + VAL "37"
```

In effetti la prima linea occupa 31 byte e la seconda ne occupa 25, ma a volte sono questi pochi byte che decidono se un programma può o meno girare sul vostro Sinclair. La linea:

```
10 LET A = VAL "25 + 37"
```

consente un ulteriore risparmio ed occupa solo 22 byte, 9 in meno rispetto ai 31 della prima linea esaminata.

Se si considera che all'accensione del computer (1 Kbyte) si dispone di 899 byte effettivamente utilizzabili (125 vengono usati dal computer) e se si pensa che la gestione dello schermo ne richiede una quantità comunque elevata, si può capire che la possibilità di risparmiare ben 9 byte in una sola linea assume un'importanza notevole.

Dunque ecco alcuni consigli per risparmiare memoria:

a) *Usate solo* le prime linee in alto dello schermo e ricordate che ogni linea usata consuma 33 byte; se riempite lo schermo vi resteranno per il programma solo (899-793) = 106 byte.
   Nel programma MASTER MIND si può notare come con l'istruzione POKE 16418, 18 si utilizzano solo le prime sei linee dello schermo, e si eseguono degli scroll che hanno effetto solo dalla linea 5 in su.
   Quando le linee di programma vengono immagazzinate occupano byte sia nella memoria del programma sia delle variabili.
   In particolare nella memoria del programma:
   – ogni linea occupa 5 byte fissi;
   – ogni numero posto fuori dalle stringhe che non sia il numero di linea occupa 6 byte, più un byte per ogni cifra che lo compone;
   – qualsiasi altro carattere, segno di punteggiatura, istruzione, funzione, contenuto di stringa, occupa 1 byte.
   Nella memoria delle variabili del programma:
   – ogni variabile numerica occupa 5 byte più 1 byte per ogni carattere nel suo nome;
   – ogni variabile di controllo dei FOR occupa 18 byte;
   – ogni variabile stringa occupa 3 byte più 1 byte per ogni carattere nella stringa.
   Considerando quanto detto:

b) *Si utilizza* la funzione VAL per indicare i numeri: "11" occupa 4 byte, 11 ne occupa 8.

c) *Si utilizzi* se possibile sempre la stessa variabile di controllo dei cicli; per esempio, FOR K quando i cicli non sono nidificati.

d) *Si usi* PI, SGN, ABS, NOT, ecc. Per esempio:
   A = 0 = 13 byte
   A = PI − PI = 11 byte

Uso degli operatori logici

Le linee:

```
10 IF A=1 THEN PRINT "A=1"
20 IF A=2 THEN PRINT "A=2"
```

possono essere sostituite da:

```
22 PRINT ("A=1" AND A=1) + ("A=2" AND
   A=2)
```

Se A è uguale a 1, la prima espressione tra parentesi nella linea 22 è vera, la seconda è falsa: la prima verrà scritta, la seconda no. Quindi se A=1 la linea 22 va interpretata:

```
22 PRINT "A=1" + " "
```

viceversa nel caso opposto. Le linee 10 e 20 occupano complessivamente 56 byte, la linea 22 ne occupa 41: 15 di meno.

Analogamente le linee:

```
10 IF H=0 THEN LET A = A+1152
20 IF G=1 THEN LET A = A+360
```

possono essere sostituite da:

```
22 LET A = A + (1152 AND H=0) + (G=1 AND
   360)
```

Se H=0 e G <> 1, avremo:

```
LET A = A + 1152 + 0
```

Se invece H=0 e G=1, si avrà:

```
LET A = A + 1152 + 360
```

Errori di logica con AND e OR

Un errore di logica molto comune con AND e OR è:

```
100 IF A=H AND B=Z OR B=W THEN
    GOTO 1000
```

Il computer salta alla linea 1000 tutte le volte che:

```
A=H e B=Z
```

inoltre tutte le volte che:

```
B=W
```

anche se A è diverso da H.

Per avere un comportamento corretto si deve scrivere:

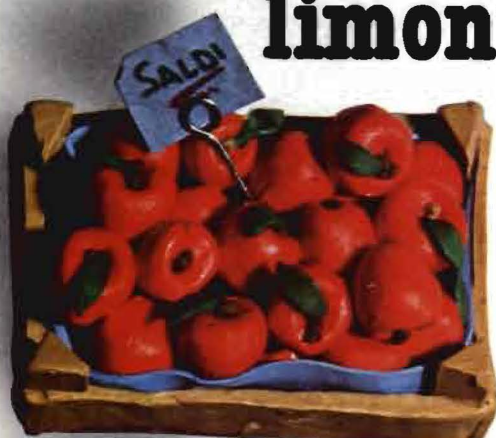
```
100 IF A=H AND B=Z OR A=H AND B=W
    THEN GOTO 1000
```

oppure

```
100 IF A=H AND (B=Z OR B=W) THEN GOTO
    1000
```

ancora oggi!

Ancora oggi c'è chi compera personal come fossero mele, arance o limoni...









...tu, invece, oggi acquista

Io oggi ho scelto MPF I
E sono soddisfatto.



MPF II l'utilizzo dappertutto. È leggero, compatto, grande come una agenda. Con lui oggi muovo i primi passi nell'affascinante mondo dell'informatica. Sono sicuro che insieme a me crescerà e sarà capace di aiutarmi domani nel mio lavoro. Un semplice video-gioco, un valido home computer, un indispensabile personal? Lo decido io! E questo mi soddisfa.

MPF II ha una struttura molto compatta e si avvale di soluzioni hardware originali ed espandibili. La più immediata è la tastiera esterna la cui connessione all'unità centrale è molto semplice.
Inoltre una serie di opzionali (disk drive, stampanti termiche, stampanti su carta normale, sintetizzatore vocale, monitor di formati diversi e con diversi tipi di fosfori, interfaccia seriale RS232C, joy-stick, generatore di suoni ed altro ancora) con i quali trasformi il tuo home computer in un personal professionale. Vuoi potenziare il tuo sistema informativo? Non devi ricominciare da capo. Sono tanti i connettori sui lati dell'MPF II che permettono di espanderlo fino a configurazioni estremamente potenti e già tutte attuabili.
Scegli tu!
Così hai la possibilità di divertirti, di studiare, di imparare il linguaggio Basic, sempre più importante. MPF II è accompagnato dai manuali d'uso e dal manuale di programmazione Basic tutti in lingua italiana. Un comodo ausilio di lavoro.
Il software è ampio e completo nelle tante cassette, nei dischi, nelle cartucce che vengono fornite insieme ad MPF II. È inoltre possibile accedere alla vasta bibliografia di programmi esistenti per la sua compatibilità di Basic...! MPF II, non scordiamolo, è dotato della tastiera incorporata e della scheda colore già installata. Tutto viene soddisfatto, i tuoi desideri, i tuoi giochi, le tue necessità, i tuoi lavori, la tua creatività. Pensa a qualcosa di grande per te, senza credere di sognare. MPF II è piccolo, leggero, ma ha grandi capacità di memoria e d'uso. Noi lo chiamiamo "l'investimento espandibile". E tu? Sceglilo e sarai al centro dell'attenzione di tutti.
Nella sua simpatica e morbida borsa da viaggio, insieme con tutti i componenti del sistema, viene sul lavoro, torna a casa, ti aiuta nello studio. Insomma MPF II è una scelta che ti dà soddisfazione, un sicuro investimento produttivo.

CPU
R 6502

ROM
16K Bytes

I. Il mio primo ed unico computer.

Caratteristiche

L'unità centrale ha una tastiera alfanumerica di 49 tasti multifunzione con i quali c'è la possibilità di generare 153 codici ASCII.
È possibile il completo controllo del cursore tramite 4 appositi tasti. Lo schermo visualizza 24 righe per 40 colonne. Lavora con un set di caratteri ASCII maiuscolo e caratteri grafici speciali (50) raggiungibili dalla tastiera tramite il CTRL-B.
È disponibile una grafica contemporanea in 2 risoluzioni, high con 280x192 punti e low con 40x48 punti, a colori. È possibile miscelare testo e grafica.
Il microprocessore è il 6502. Sulla ROM è disponibile l'interprete Basic ed un monitor con disassemblatore per programmare anche in linguaggio macchina. L'altoparlante è presente.
L'unità centrale ha ben 64 K di memoria RAM dinamica e 16 K ROM. L'apposito slot porta all'esterno il BUS dati e indirizzi oltre ai segnali di controllo di tutto il computer. È possibile collegare interfacce e periferiche di tipo più svariato. L'unità centrale viene già fornita con un interfaccia parallela per stampanti entro contenuta.



MICRO-PROFESSOR MPF II
l'investimento espandibile

RAM
64K Bytes

Interprete Basic
più di 90
istruzioni

Scrivici per ulteriori informazioni e per sapere dove puoi trovare MPF II vicino a casa tua.
PS 83

Nome ____________________

Cognome ____________________

Indirizzo ____________________

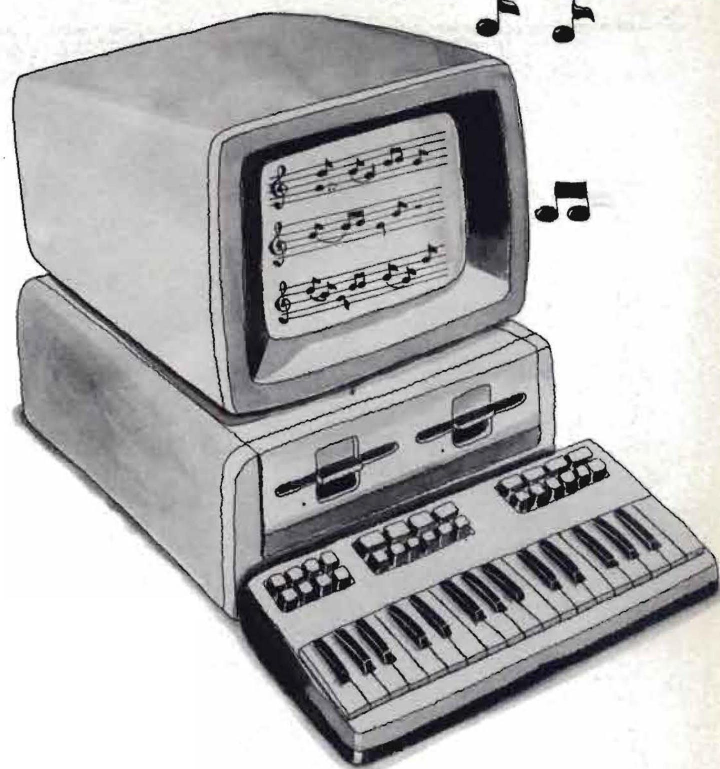
____

____

DIGITEK COMPUTER

Ufficio Vendite
Via Marmolada, 9/11 43058 SORBOLO (Parma)
Tel. 0521/69635 Telex 531083

Ag. advert Reggio E.



Musica con TI99/4A

Abbiamo ricevuto da Dario Paganini di Varese la conversione per TI99/4A del programma "Comporre musica con l'Atari" pubblicato sul n. 5. Il funzionamento è analogo a quello del programma originale.

Le linee da 10 a 80 stabiliscono (come da tabella) le corrispondenze fra i tasti e la frequenza delle note, usando le istruzioni READ/DATA.

I passi 1000/1110 sono relativi alla solita CALL KEY. Se non è stato premuto nessun tasto il puntatore S è 0 e viene riletta la tastiera. Se si è premuto un tasto corrispondente ad una nota, il programma pone il valore opportuno nella variabile TONE e (piccola modifica rispetto al programma Atari) riproduce quest'ultima al passo 1105 con la CALL SOUND. Se si è premuto un tasto richiedente un accordo il programma va alla routine che lo forma.

Ho aggiunto un confronto IF-THEN ai passi 2000/2100/2200/2300 per evitare che il programma si fermasse nel caso che il primo tasto premuto fosse un richiamo ad una routine per la formazione di un accordo. In questo caso infatti essendo la variabile TONE = 0 la CALL SOUND non potrebbe essere eseguita in quanto leggerebbe una frequenza 0 e il calcolatore darebbe il messaggio di errore "BAD VALUE IN...".

I valori di durata e volume della CALL SOUND possono essere facilmente variati secondo il gusto personale.

NOTA	TASTO	ASCII	FREQUENZA
LA	N	78	440
LA	J	74	466
SI	M	77	494
DO	Z	90	262
DO	S	83	277
RE	X	88	294
RE	D	68	311
MI	C	67	330
FA	V	86	349
FA	G	71	370
SOL	B	66	392
SOL	H	72	415
maggiore	1	49	—
settima	2	50	—
minore	3	51	—
min. 7ª	4	52	—

Tabella 1. *Corrispondenze fra tasti, codici ASCII, frequenze.*

```
10   DIM REALTONE (12,2)
20   LASTBYTE=0
30   FOR I=1 TO 12
40   READ A,B
50   REALTONE (I,1)=A
60   REALTONE (I,2)=B
70   NEXT I
80   DATA 78,440,74,466,77,494,90,262,
          83,277,88,294,68,311,67,330,
          86,349,71,370,66,392,72,415
1000 CALL KEY(0,K,S)
1010 IF S=0 THEN 1000
1020 IF K=49 THEN 2000
1030 IF K=50 THEN 2100
1040 IF K=51 THEN 2200
1050 IF K=52 THEN 2300
1060 LASTBYTE=K
1070 FOR I=1 TO 12
1080 IF K<>REALTONE(I,1) THEN 1100
1090 TONE=REALTONE(I,2)
1100 NEXT I
1105 CALL SOUND(300,TONE,5)
1110 GOTO 1000
2000 IF TONE=0 THEN 1000
2010 CALL SOUND(300,TONE,5,INT(TONE/
     .79394+.5),5,INT(TONE/.66837+.5),5)
2020 GOTO 1000
2100 IF TONE=0 THEN 1000
2110 CALL SOUND(300,TONE,5,INT(TONE/
     .79394+.5),5,INT(TONE/.5625+.5),5)
2120 GOTO 1000
2200 IF TONE=0 THEN 1000
2210 CALL SOUND(300,TONE,5,INT(TONE/
     .84244+.5),5,INT(TONE/.66837+.5),5)
2220 GOTO 1000
2300 IF TONE=0 THEN 1000
2310 CALL SOUND(300,TONE,5,INT(TONE/
     .84244+.5),5,INT(TONE/.5625+.5),5)
2320 GOTO 1000
2400 END
```

Labirinto per Spectrum

Il lettore Michele Bighignoli di Palermo ha convertito per lo ZX-Spectrum il programma per la generazione casuale di labirinti presentato sul numero 3 del Dicembre 1982.

Il programma pubblicato disegna un labirinto privo di ingresso ed uscita e con stampato sul percorso il parametro casuale 'j' relativo alla direzione presa nel tracciare il labirinto. Sono dati necessari all'algoritmo e non possono essere omessi, ma possono essere resi invisibili sostituendo alla linea 160 'INK 1'; con 'INK 7'; e premettendo lo stesso comando al "5" della linea 140. I segmenti bianchi ed i neri sono tutti spazi con diverso colore di PAPER, non è quindi possibile trasferire il labirinto su carta con il comando COPY.

```
 100 DIM a(4): LET a(1)=2: LET a
(2)=-2: DIM b(4): LET b(3)=2: LE
T b(4)=-2
 110 INK 0: PAPER 7: BORDER 7: O
VER 0: CLS
 120 FOR f=0 TO 20: FOR e=0 TO 3
0: PRINT AT f,e; PAPER 0;" ": NE
XT e: NEXT f
 130 LET a=1: LET b=1
 140 PRINT AT a,b;"5"
 150 LET j=INT (RND*4)+1: LET x=
j
 160 IF ATTR (a+a(j),b+b(j))=0 T
HEN PRINT AT a+a(j),b+b(j); INK
1;j;AT a+a(j)/2,b+b(j)/2;" ": LE
T a=a+a(j): LET b=b+b(j): GO TO
150
 170 LET j=j*(j<4)+1: IF j<>x TH
EN GO TO 160
 180 LET j=VAL (SCREEN$ (a,b)):
PRINT AT a,b;"": IF j<5 THEN LET
 a=a-a(j): LET b=b-b(j): GO TO 1
50
```

Rally per ZX 81

Sul numero 3 avevamo pubblicato il programma Rally per computer Atom. Marcello Morchio di Genova ne ha fatto una "sintesi", più che una conversione, per ZX 81.

Confrontando il programma originale ci si può rendere conto di come poche funzioni in più possono complicare la programmazione.

Alcune osservazioni:

1) è necessario stampare il doppio bordo "oo" perché se lo si stampa semplice ("o") la macchina può sfuggire in diagonale;
2) le linee 85 e 86 controllano che la strada non si sposti fuori schermo;
3) la linea 105 disattiva i controlli sulla macchina fino a quando non si è "in strada";
4) la pista è diversa ogni volta. Può essere resa sempre uguale inserendo una linea del tipo 5 RAND n con n che alternino il percorso.

```
   1 SLOW
  10 CLS
  15 PRINT "LARG.?"
  16 INPUT K
  17 IF K<3 OR K>30 THEN GOTO 15
  20 LET Y=2
  25 LET S=0
  30 LET T=0
  40 LET O=20
  50 LET A$="OO"
  60 PRINT AT 21,T;A$
  70 PRINT AT 21,T+K;A$
  80 LET T=T+INT (RND*3)-1
  85 IF T<0 THEN LET T=T+1
  66 IF T+K>30 THEN LET T=T-1
  90 PRINT AT 10,Y;" "
 100 SCROLL
 102 LET S=S+1
 105 IF S<10 THEN GOTO 130
 110 IF INKEY$="8" THEN LET Y=Y+
1
 120 IF INKEY$="5" THEN LET Y=Y-
1
 130 PRINT AT 10,Y;
 150 IF PEEK (PEEK 16398+256*PEE
K 16399)=CODE A$ THEN GOTO 190
 170 PRINT "0"
 180 GOTO 60
 190 PRINT AT 10,Y-2;".:*:."
 200 PRINT "KM:";INT (S/K),"GG:"
;INT (S/48)
 210 PAUSE 4E4
 220 RUN
```

Project-Robot

Questo programma è la versione VIC 20 di Project-Robot, pubblicato sul n. 7.

Funziona allo stesso modo del programma originale: non ho aggiunto né colori né suoni par adattarmi ai "3583 byte free" di base, ma chi ha un'espansione può rendere il programma più appariscente.

Ecco le principali modifiche apportate:

1) non è ammesso, nel BASIC VIC, scrivere "GOSUB variabile", quindi le tre routine di Laser, Radar, Muovi vengono richiamate con GOSUB 500, GOSUB 650, GOSUB 750 e le variabili omonime sono soppresse;
2) il VIC riconosce solo le prime due lettere delle variabili. Quindi, le variabili sono state così cambiate: Display→DF, Danni→DN, Azioni→AZ, Posx→PX, Posy→PY, Cont→CT, Poslaser→PL, Eco→EC, Poseco→PE.

Sono rimaste inalterate le variabili K, R e gli array Z e Z$.

Il programma lascia liberi, tolte le variabili, circa 250 byte; c'è quindi un certo margine per sostituire le routine di movimento dei robot con altre più lunghe. Queste potranno usare le variabili ZA, ZB, ..., ZM inizializzate ma non usate dal programma principale.

Per ulteriori istruzioni, vedere l'articolo originale.

```
10 ZA=0:ZB=0:ZC=0:ZD=0:ZE=0:ZF=0:ZG=0:ZH=0:ZI=0::ZJ=0:ZK=0:ZL=0:OU=0:PRINT"J";:G
OTO100
50 FORK=1TO3:PRINT"ROBOT N."K;
51 E$="":INPUTE$:IFLEN(E$)<1ORLEN(E$)>10THENPRINT"■ERRORE!!!":GOTO51
52 Z$(K)=E$:NEXT:FORK=1TO3:PRINT"■"Z$(K):Z(K,5)=PEEK(DF):NEXT
70 PRINT"JQQQQQQQQQQQ********************** *>> PROJECT ROBOT <<* ************
*********
72 PRINT"IN GARA:":FORK=1TO3:PRINTZ$(K):NEXT:RETURN
100 DIMZ(3,5):DIMZ$(3):NL=1:DF=7680:N=-22:O=1:S=22:E=-1:NO=-21:SO=23:SE=21:NE=-2
3
160 FORK=1TO3:Z(K,2)=N:Z(K,3)=0:NEXT:GOSUB50
190 FORK=1TO3:Z(K,1)=DF+INT(RND(1)*10+1)+INT(RND(1)*10+1)*22:NEXT
250 PRINT"██████████████":FORI=1TO11:PRINT"■                    ■":NEXT:PRINT"██████████████
300 FORR=1TO3:IFZ(R,3)>3THEN400
310 POKEZ(R,1),Z(R,5):DR=Z(R,2):DN=Z(R,3):AZ=3:Q=Z(R,1)-DF:PY=INT(Q/22):PX=Q-PY*
22
350 ONRGOSUB7000,8000,9000
355 Z(R,2)=DR
360 NEXTR
365 GOTO300
400 IFZ(R,3)=5THEN360
410 Z(R,3)=5:PRINT"■QQQQ"SPC(22*R+50)SPC(250)"■OUT!!!":OU=OU+1:IFOU<2THEN360
420 FORR=1TO3:IFZ(R,3)=5THENNEXT
430 FORI=1TO1000:NEXT:PRINT"JQQQQHA VINTO "Z$(R):END
500 REM LASER
505 IFAZ=0THENRETURN
510 CT=0:AZ=AZ-1:PL=Z(R,1)+DR
540 IFPEEK(PL)<>32THEN570
545 POKEPL,46:FORI=1TO50:NEXT:CT=CT+1:PL=PL+DR:POKEPL-DR,32:GOTO540
570 IFPEEK(PL)<>102THEN585
575 Z(R,4)=Z(R,4)-CT:RETURN
585 M=PEEK(PL):POKEPL,32:IFM<>Z(1,5)THEN610
600 Z(1,3)=Z(1,3)+1:GOTO575
610 IFM<>Z(2,5)THEN625
615 Z(2,3)=Z(2,3)+1:GOTO575
625 Z(3,3)=Z(3,3)+1:GOTO575
650 REM RADAR
655 IFAZ=0THENRETURN
660 CT=0:EC=0:AZ=AZ-1:PE=Z(R,1)+DR
680 IFPEEK(PE)<>32THEN700
685 CT=CT+1:PE=PE+DR:GOTO680
700 IFPEEK(PE)=102THENRETURN
705 EC=CT:RETURN
750 REM MUOVI
755 IFAZ=0THENRETURN
760 AZ=AZ-1:IFPEEK(Z(R,1)+DR)<>32THENRETURN
770 POKEZ(R,1),32:Z(R,1)=Z(R,1)+DR:POKEZ(R,1),Z(R,5):Q=Z(R,1)-DF:PX=INT(Q/22):PY
=Q-22*PX
800 RETURN
5000 REM ROBOT:           1=RIGA 7000           2=RIGA 8000           3=RIGA 90
00
7000 REM SFONDO
7005 IFPY=11THEN7200
7010 DR=S:GOSUB750:GOSUB750:GOSUB750:RETURN
7100 DR=O:GOSUB750:DR=N:GOSUB650:IFEC<>0THENGOSUB500
7160 DR=O:IFPX=10THENDR=E
7165 RETURN
7200 IFDR=OTHEN7100
7210 DR=E:GOSUB750:DR=N:GOSUB650:IFEC<>0THENGOSUB500
7260 DR=E:IFPX=1THENDR=O
7270 RETURN
8000 REM ESPLORATORE

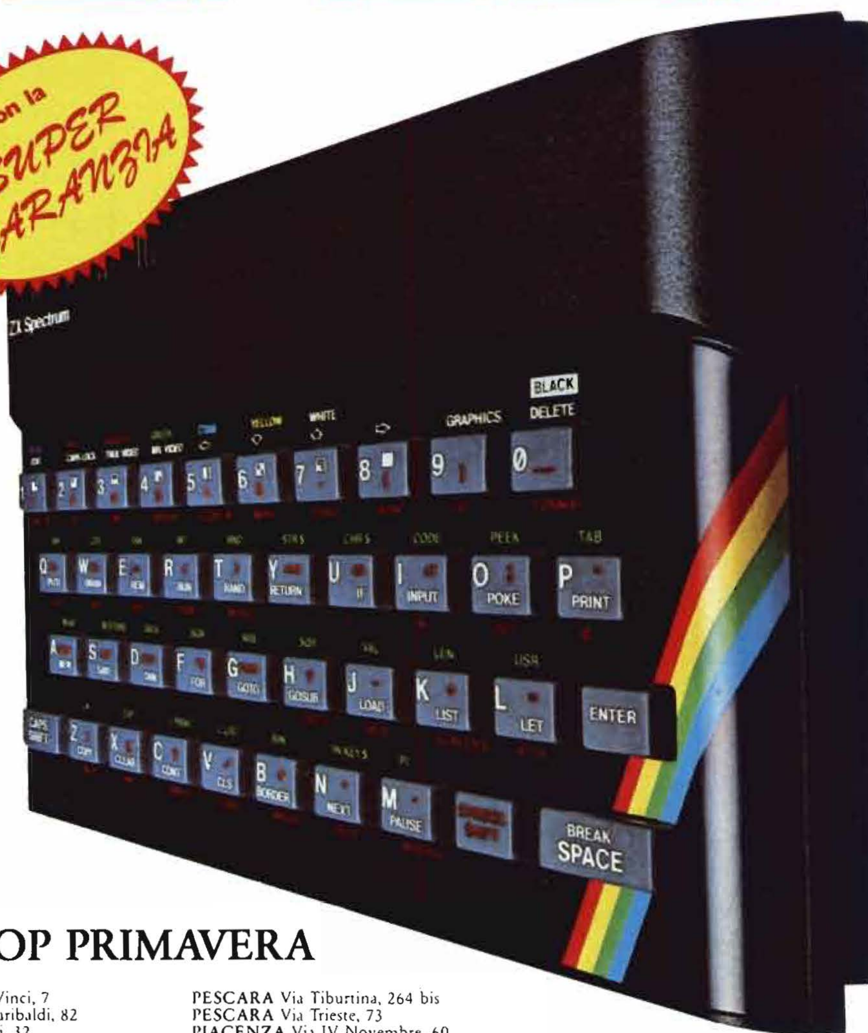
8005 IFZA=1THEN8065
8010 W=INT(RND(1)*8)
8025 IFW=1THENDR=NO
8030 IFW=2THENDR=O
8035 IFW=3THENDR=SO
8040 IFW=4THENDR=S
8045 IFW=5THENDR=SE
8050 IFW=6THENDR=E
8055 IFW=7THENDR=NE
8060 IFW=0THENDR=N
8065 GOSUB650:ZA=0:IFEC<>0THENZA=1:GOSUB500
8085 IFEC=0THENGOSUB750
8090 RETURN
9000 REM ANGOLO
9005 IFPX=1ANDPY=1THEN9040
9010 DR=E
9015 IFPX<>1THENGOSUB750
9020 IFPX<>1THENGOSUB750
9025 DR=N
9030 IFPY<>1THENGOSUB750
9035 IFPY<>1THENGOSUB750
9038 RETURN
9040 DR=S:GOSUB650:IFEC<>0THENGOSUB500
9050 DR=O:GOSUB650:IFEC<>0THENGOSUB500
9055 RETURN
```

QUI C'E' sinclair

ZX Spectrum 16/48 k RAM.

- 16 o 48 kbytes RAM.
- grafica ad alta risoluzione (256x192 punti).
- 8 colori da utilizzare con la più assoluta libertà per testo, sfondo, bordo, in campo diretto o inverso, con due gradi di luminosità, a luce fissa o lampeggiante.
- Tastiera multifunzione con maiuscole, minuscole, simboli grafici, caratteri definibili dall'utente.
- BASIC Sinclair esteso con funzioni a un tasto per programmare in fretta e senza errori.
- Funzioni specifiche per la grafica e per la gestione di dati d'archivio.
- Ampia disponibilità di programmi preregistrati su compact-cassette: giochi, passatempi, educazionali, matematici, gestionali.
- Totale compatibilità con la stampante ZX.
- Disponibilità immediata del volume ALLA SCOPERTA DELLO ZX SPECTRUM in italiano.
- Prezzo eccezionale: 299.000 lire nella versione a 16 kbytes.





Lo trovi anche nel tuo BIT SHOP PRIMAVERA

AGRATE BRIANZA Via G. Matteotti, 99
ALBA Via Paruzza, 2
ALESSANDRIA Via Savonarola, 13
ANCONA Via De Gasperi, 40
AOSTA Av. Conseil Des Commis, 16
BARI Via Capruzzi, 192
BASSANO DEL GRAPPA Via Jacopo Da Ponte, 51
BERGAMO Via S. F. D'Assisi, 5
BIELLA Via Italia, 50A
BRESCIA Via B. Croce, 11/13/15
BUSTO ARSIZIO Via Gavinana, 17
CAGLIARI Via Zagabria, 47
CALTANISSETTA Via R. Settimo, 10
CAMPOBASSO Via Mons. II Bologna, 10
CATANIA Via Muscatello, 6
CESANO MADERNO Via Ferrini, 6
CESENA Via F.lli Spazzoli, 239
CINISELLO BALSAMO V.le Matteotti, 66
COMO Via L. Sacco, 3
COSENZA Via Dei Mille, 86
CREMA Via IV Novembre, 56/58
CUNEO C.so Nizza, 16
FAVRIA CANAVESE C.so G. Matteotti, 13
FIRENZE Via G. Milanesi, 28/30
FORLÌ P.zza Melozzo, 1
GALLARATE Via A. Da Brescia, 2
GENOVA Via Domenico Fiasella, 51/R
GENOVA C.so Gastaldi, 77/R
GENOVA-SESTRI Via Chiaravagna, 10/R
GENOVA-SESTRI Via Ciro Menotti, 136/R
IMPERIA Via Delbecchi, 32
LATINA Via E. Toti (Galleria Cisa)
LECCE V.le Marche, 21
LECCO Via L. Da Vinci, 7
LEGNANO C.so Garibaldi, 82
LIVORNO Via Paoli, 32
LODI V.le Rimembranze, 36/B
LUCCA Via S. Concordio, 160
LUGO (RA) Via Magnapassi, 26
MACERATA Via Spalato, 126
MANTOVA Via Cavour, 69
MERANO Via S. Maria del Conforto, 22
MESSINA Via Del Vespro, 71
MESTRE P.zza Ferretto, 78
MILANO Via G. Cantoni, 7
MILANO Via E. Petrella, 6
MILANO Via Altaguardia, 2
MILANO P.zza Firenze, 4
MILANO V.le Corsica, 14
MILANO V.le Certosa, 91
MILANO Galleria Manzoni, 40
MIRANO-VENEZIA Via Gramsci, 40
MODENA Via Fonteraso, 18
MONZA Via Azzone Visconti, 39
MORBEGNO Via Fabani, 31
NAPOLI Via Luigia Sanfelice, 7/A
NAPOLI C.so Vittorio Emanuele, 54
NAPOLI Via Luca Giordano, 40/42
NOVARA Baluardo Q. Sella, 32
NOVARA Via Perazzi, 23/B
PADOVA Via Fistomba, 8 (Stanga)
PADOVA Via Piovese, 37
PALERMO Via Libertà, 191
PARMA Via Imbriani, 41
PAVIA Via C. Battisti, 4/A
PERUGIA Via R. D'Andreotto, 49/55
PESCARA Via Tiburtina, 264 bis
PESCARA Via Trieste, 73
PIACENZA Via IV Novembre, 60
PISA Via Emilia, 36
PISA Via XXIV Maggio, 101
PISTOIA V.le Adua, 350
POMEZIA Via Roma, 39
POTENZA Via G. Mazzini, 72
POZZUOLI Via G.B. Pergolesi, 13
PRATO Via E. Boni, 76/78
RECCO Via B. Assereto, 78
RIMINI Via Bertola, 75
ROMA P.zza San Donà Di Piave, 14
ROMA V.le IV Venti, 152
ROMA Via Cerreto Da Spoleto, 23
ROMA Via Ponzio Cominio, 46
ROMA Via Del Traforo, 136
ROMA Via G. Villani, 24-26
S. DONÀ DI PIAVE P.zza Rizzo, 61
SASSUOLO P.zza Martiri Partigiani, 31
SAVONA Via G. Scarpa, 13/R
SENIGALLIA Via Maierini, 10
SONDRIO Via N. Sauro, 28
TERAMO Via Martiri Pennesi, 14
TORINO C.so Grosseto, 209
TORINO Via Tripoli, 179
TORINO Via Nizza, 91
TRENTO Via Sighele, 7/1
TRIESTE Via Fabio Severo, 138
TRIESTE Via Torrebianca, 18
TRIESTE Via Madonna del Mare, 7
UDINE Via Tavagnacco, 89/91
VARESE Via Carrobbio, 13
VENEZIA Cannaregio, 5898
VERCELLI Via Dionisotti, 18
VIAREGGIO Via A. Volta, 79
VICENZA Via del Progresso, 7/9
VIGEVANO C.so V. Emanuele, 82
VOGHERA P.zza G. Carducci, 11



La più grande catena di computer in Europa.

Rotazione bidirezionale sul video dello ZX81

La risposta ai molti lettori che vogliono sapere come si può ottenere lo scorrimento laterale del video dello ZX81, ci è arrivata da Carlo Lagomarsini e Alberto Muzio di Arcola (La Spezia), che hanno raccolto la richiesta di Marcello Morchio pubblicata sul numero 5. Ispirandosi ad un articolo di Antonio Pierini apparso su un'altra rivista, hanno realizzato due routine in linguaggio macchina per ruotare con ricircolo, come direbbe chi ha studiato un po' di Assembler, una finestra rettangolare. La cosa più interessante è che posizione e dimensione, da due caratteri, a tutto il video, sono definibili tramite un inizializzatore BASIC.

Noi della Redazione abbiamo aggiunto un semplice caricatore, un dumper per verificare il contenuto della memoria e abbiamo messo tutto insieme in un unico programma, comprese le routine già caricate nelle REM iniziali... ma facciamo un po' d'ordine:

linea 1; linguaggio macchina per la rotazione verso sinistra (BASIC dalla linea 9300), indirizzo di lancio 16514, non è rilocabile, gli zeri in fondo sono indispensabili per avere gli indirizzi giusti nell'altra routine: devono essere esattamente 22;

linea 2; linguaggio macchina per la rotazione verso destra (BASIC dalla linea 9000), indirizzo di lancio 16573, non è rilocabile;

linee 10-60; caricatore (opzionale), si lancia con RUN 10. Per caricare le routine si può provare a scrivere direttamente le REM come appaiono nel listato oppure prepararle con un numero sufficiente di caratteri qualsiasi e far girare questo caricatore dando i numeri del CECK DUMP pubblicato. Se si usa il secondo metodo si faccia attenzione che la prima REM abbia (esattamente) 53 caratteri;

linee 1000-1070; dumper (opzionale), lista le due routine, si attiva con RUN 1000;

linee 8000-8100; master che gestisce le routine in BASIC e in linguaggio macchina, deve essere sostituito dal programma utilizzatore;

linee 9000-9230; inizializzatore della seconda routine, una volta usato può essere cancellato;

linee 9300-9410; inizializzatore della prima routine, una volta usato può essere cancellato.

Non ci dilunghiamo in spiegazioni sul linguaggio macchina, ma pubblichiamo i listati Assembler delle routine e il "programmone" BASIC, sezionabile secondo le esigenze individuali. Provate e buon divertimento...

```
1234567890123456789012345678901 2
1234567890123456789012345678901 2
 SCEGLI LA FUNZIONE: 23456789012
1234567890123456789012345678901 2
 1) DEFINIZIONE FINESTRA <<< 012
 2) DEFINIZIONE FINESTRA >>> 012
 3) SCORRIMENTO <<< 1234567890 12
 4) SCORRIMENTO >>> 1234567890 12
12345678901234567890123456789012
12345678901234567890123456789012
12345678901234567890123456789012
12345678901234567890123456789012
12345678901234567890123456789012
12345678901234567890123456789012
12345678901234567890123456789012
12345678901234567890123456789012
12345678901234567890123456789012
12345678901234567890123456789012
12345678901234567890123456789012
12345678901234567890123456789012
12345678901234567890123456789012
12345678901234567890123456789012
```

Figura 1. *Copy del quadro iniziale. Lo spostamento dei numeri permette di evidenziare le finestre definite.*

```
CECK DUMP
16514 = 42      16580 = 35
16515 = 12      16581 = 84
16516 = 64      16582 = 93
16517 = 1       16583 = 6
16518 = 213     16584 = 31
16519 = 2       16585 = 126
16520 = 9       16586 = 35
16521 = 22      16587 = 78
16522 = 22      16588 = 119
16523 = 229     16589 = 121
16524 = 6       16590 = 16
16525 = 31      16591 = 250
16526 = 126     16592 = 18
16527 = 43      16593 = 205
16528 = 78      16594 = 228
16529 = 119     16595 = 64
16530 = 121     16596 = 62
16531 = 16      16597 = 22
16532 = 250     16598 = 61
16533 = 6       16599 = 40
16534 = 2       16600 = 5
16535 = 43      16601 = 50
16536 = 16      16602 = 213
16537 = 253     16603 = 64
16538 = 193     16604 = 24
16539 = 2       16605 = 230
16540 = 21      16606 = 62
16541 = 194     16607 = 22
16542 = 139     16608 = 50
16543 = 64      16609 = 213
16544 = 201     16610 = 64
                16611 = 201
16573 = 42      16612 = 6
16574 = 12      16613 = 1
16575 = 64      16614 = 35
16576 = 17      16615 = 16
16577 = 0       16616 = 253
16578 = 0       16617 = 201
```

Figura 2. *Dump delle routine già inizializzate per lo scorrimento di tutto il video. Se avete battuto anche il dumper (ed avete la stampante collegata) si ottiene con RUN 1000.*



ce l'hai?

Il tuo Spectrum è preziosissimo difendilo con la "SUPER GARANZIA"
La Rebit Computer, distributore per l'Italia dei prodotti **SINCLAIR**, ha messo a punto la nuova straordinaria

SUPER GARANZIA

Apri la scatola del tuo **SPECTRUM** acquistato presso un Rivenditore Autorizzato e ci trovi anche un libretto: ti accompagnerà nei tuoi futuri acquisti, dandoti l'occasione per risparmiare oltre 100.000 lire. Ti darà la Garanzia di una perfetta assistenza, e avrai la certezza del valore del tuo autentico **SPECTRUM**. Il libretto della "SUPER GARANZIA" contiene le modalità per l'iscrizione al **SINCLUB**, la federazione di tutti i Sinclair Club Italiani. Inoltre il Coupon sconto per abbonarsi a **"SPERIMENTARE"** il mensile di elettronica che pubblica il bollettino Sinclub: idee, programmi, notizie, vita associativa.

La tessera Software ti da diritto ad uno sconto sull'acquisto dei programmi. Infine nel libretto "SUPER GARANZIA" troverai la possibilità di acquistare la stampante **ZX PRINTER SINCLAIR** ad un prezzo eccezionale.

PER QUESTO UNO SPECTRUM SENZA LA "SUPER GARANZIA" E' SOLO UN MEZZO
Spectrum





sinclair

Spectrum

molto di più di una garanzia!!



```
   1 REM E£RND▘STR$ ▘-- FAST ▞3
 IF ▞▘F( CLEAR AT ▘+TAB █RNDTAN
0000000000000000000000
   2 REM E£RND) ;???▞3 IF >LN S
LOW RNDY-XC▙MSTR$ RND/ NEW Y-MST
R$ RNDTAN ▞ 7( CLEAR TAN 0000000
0000000000000000000000000000
   3 GOTO 8000
  10 REM CARICATORE
  11 FAST
  12 LET B=16514
  13 LET E=16544
  14 GOSUB 20
  15 LET B=16573
  16 LET E=16617
  17 GOSUB 20
  18 STOP
  20 FOR A=B TO E
  25 PRINT A;" = ";
  30 INPUT D
  35 PRINT D
  40 POKE A,D
  50 NEXT A
  60 RETURN
1000 REM DUMPER
1001 FAST
1004 LPRINT
1005 LPRINT "CECK DUMP"
1006 LPRINT
1010 FOR I=16514 TO 16544
1020 LPRINT I;" = ";PEEK I
1030 NEXT I
1035 LPRINT
1040 FOR I=16573 TO 16617
1050 LPRINT I;" = ";PEEK I
1060 NEXT I
1070 STOP
8000 REM MASTER SCORRIMENTO
8001 SLOW
8005 CLS
8010 FOR I=1 TO 22
8011 PRINT "12345678901234567890
123456789012"
8012 NEXT I
8015 PRINT AT 1,0
8020 PRINT " SCEGLI LA FUNZIONE:
"
8025 PRINT
8030 PRINT " 1) DEFINIZIONE FINE
STRA <<< "
8035 PRINT " 2) DEFINIZIONE FINE
STRA >>> "
8040 PRINT " 3) SCORRIMENTO <<<
"
8045 PRINT " 4) SCORRIMENTO >>>
"


8050 INPUT N
8060 IF N=1 THEN GOSUB 9300
8070 IF N=2 THEN GOSUB 9000
8080 IF N=3 THEN RAND USR 16514
8090 IF N=4 THEN RAND USR 16573
8095 IF N=3 OR N=4 THEN GOTO 805
0
8100 GOTO 8000
9000 REM DEFINIZIONE FINESTRA>>>
9010 CLS
9050 PRINT "SCRIVI LE COORDINATE
 DEL QUADRA-TINO IN ALTO A SINIS
TRA:"
9055 PRINT
9056 PRINT "SCRIVI X:"
9060 INPUT X1
9065 IF X1>21 OR X1<0 THEN GOTO
9056
9070 PRINT
9071 PRINT "SCRIVI Y:"
9073 INPUT Y1
9075 IF Y1>30 OR Y1<0 THEN GOTO
9071
9080 LET B=Y1+X1*33
9090 LET B2=INT (B/256)
9100 LET B1=B-B2*256
9110 POKE 16577,B1
9120 POKE 16578,B2
9125 CLS
9130 PRINT AT 0,0;"ALTEZZA DELLA
 FINESTRA ?"
9140 INPUT X
9145 CLS
9150 IF X1+X>22 THEN GOTO 9130
9155 IF X<1 THEN GOTO 9130
9160 PRINT AT 0,0;"LARGHEZZA DEL
LA FINESTRA ?"
9170 INPUT Y
9180 IF Y+Y1>32 THEN GOTO 9170
9185 IF Y<2 THEN GOTO 9160
9190 POKE 16584,Y-1
9200 POKE 16597,X
9210 POKE 16607,X
9220 POKE 16613,32-PEEK 16584
9230 CLS
9240 RETURN
9300 REM DEFINIZIONE FINESTRA<<<
9305 CLS
9310 PRINT "SCRIVI LE COORDINATE
 DEL QUADRA-TINO IN BASSO A DEST
RA:"
9315 PRINT
9316 PRINT "SCRIVI X:"
9320 INPUT X1
9325 IF X1>21 OR X1<0 THEN GOTO
9316
9330 PRINT
9331 PRINT "SCRIVI Y:"
9332 INPUT Y1
9335 IF Y1>31 OR Y1<1 THEN GOTO
9331
9340 CLS
9345 LET B=1+Y1+X1*33
9350 POKE 16518,B-INT (B/256)*25
6
9355 POKE 16519,INT (B/256)
9360 PRINT AT 0,0;"ALTEZZA DELLA
 FINESTRA ?"
9365 INPUT X
9370 IF X>22 OR X<1 OR X>X1+1 TH
EN GOTO 9365
9375 CLS
9380 POKE 16522,X
9385 PRINT AT 0,0;"LARGHEZZA DEL
LA FINESTRA ?"
9390 INPUT Y
9395 IF Y>32 OR Y<2 OR Y>Y1+1 TH
EN GOTO 9390
9396 POKE 16525,Y-1
9400 POKE 16534,34-Y
9410 CLS
9420 RETURN
```

Locaz.	Assembler	HEX	DEC	Note
16514	LD HL (16396)	2A, 0C, 40	42, 12, 64	
16517	LD, BC, nn, nn	01, nn nn	1, nn, nn	spiazzamento ultimo byte
16520	ADD HL, BC	09	9	
16521	LD D, n	16, n	22, n	n. di righe
16523	PUSH HL	E5	229	
16524	LD B, n	06, n	6, n	larghezza - 1
16526	LD A, (HL)	7E	126	
16527	DEC HL	2B	43	
16528	LD C, (HL)	4E	78	
16529	LD (HL), A	77	119	
16530	LD A, C	79	121	
16531	DJ NZ, -6	10, FA	16, 250	
16533	LD B, n	06, n	6, n	3A - larghezza
16535	DEC HL	2B	43	
16536	DJ NZ, -03	10, FD	16, 253	
16538	POP BC	C1	193	
16539	LD (BC), A	02	2	
16540	DEC D	15	21	
16541	JP NZ, (16523)	C2, 8B, 40	194, 139, 64	
16544	RET	C9	201	

Tabella 1. *Rotazione verso sinistra.*

Tabella 2. *Rotazione verso destra.*

Locaz.	Assembler	HEX	DEC	Note
16573	LD HL, (16396)	2A, 0C, 40	42, 12, 64	
16576	LD DE, nn	23, nn, nn	17, nn, nn	spiazzamento primo byte
16579	ADD HL, DE	19	25	
16580	INC HL	23	35	
16581	LD D, H	54	84	DE ← HL
16582	LD E, L	5D	93	
16583	LD B, n	06, n	6, n	larghezza -1
16585	LD A, (HL)	7E	126	
16586	INC HL	23	35	loop di riga
16587	LD C, (HL)	4E	78	
16588	LD (HL), A	77	119	
16589	LD A, C	79	121	
16590	DJ NZ, -6 ↑	10, FA	16, 250	
16592	LD (DE), A	12	18	
16593	CALL 16612	CD, E4, 40	205, 228, 64	
16596	LD A, n	3E, n	62, n	numero linee
16598	DEC A	3D	61	cont. righe
16599	JR Z, 5 ↓ (16606)	28, 05	40, 5	

Seguito tabella 2.

Locaz.	Assembler	HEX	DEC	Note
16601	LD (16597), A	32, 15, 40	50, 213, 64	decrementa il contatore di riga
16604	JR -26 ↑ (16580)	18, E6	24, 230	
16606	LD A, n	3E, n	62, n	n. linee
16608	LD (16597), A	32, 15, 40	50, 213, 64	resetta il cont.
16611	RET. (BASIC)	C9	201	
16612	LD B, n	06, n	6, n	33 - larghezza
16614	INC HL	23	35	loop di riga
16615	DJ NZ -3 ↑	22, FD	16, 253	
16617	RET (16596)	C9	201	

Operatori logici per TI99/4A

Il signor Giuseppe Devoto di Ferrara ci manda questa nota su un metodo da lui escogitato.

Si tratta di un metodo per sopperire alla mancanza degli operatori logici che non sono presenti nel linguaggio base del calcolatore.

Il metodo che ho usato, non interviene direttamente sul linguaggio con procedure hardware, ma si basa su due proprietà:

1) il computer associa alle espressioni relazionali i valori numerici 0 e −1, rispettivamente, a seconda che siano "false" o "vere";
2) gli statement IF A<>O THEN ... e IF A THEN vengono considerati identici.

Vediamo come, sfruttando queste due proprietà, si possano realizzare dei surrogati degli operatori logici fondamentali, e cioè: And, Or e Not.

Iniziamo con l'operatore And, definito secondo la figura 1.

Il metodo che io utilizzo consiste nello scrivere, ad esempio, IF (A>B) * (B<C) THEN... che risulta equivalente a IF A>B AND B<C THEN...

Infatti: l'enunciato IF... THEN... esegue il salto condizionato alla riga indicata soltanto se l'espressione (A>B) * (C<D) fornisce un risultato diverso da zero, una volta che alle espressioni relazionali si sia sostituito il valore numerico corrispondente.

Se entrambe le relazioni sono "vere", il computer associa il valore numerico −1 a ciascuna di esse e il risultato complessivo è 1.

Se, invece, una delle due è falsa, il computer vi associa il valore 0 e il risultato complessivo è 0.

Nel primo caso il salto condizionato viene eseguito, nel secondo no.

Questo è esattamente il funzionamento richiesto per l'operatore logico And.

Si noti che non c'è alcun limite al numero di espressioni relazionali che l'operatore può legare; si può infatti scrivere: IF (A>B) * (B<C) * ... * (D=F) THEN...

Anche in questo caso il salto condizionato viene eseguito solo se sono vere tutte le espressioni indicate.

Per l'operatore Or, definito secondo la figura 2, si procede seguendo lo stesso principio.

Lo statement:

```
IF (A>B) + (C<D) THEN...
```

sostituisce:

```
IF A>B OR C<D THEN...
```

x	y	x and y
Vera	Vera	Vera
Vera	Falsa	Falsa
Falsa	Vera	Falsa
Falsa	Falsa	Falsa

Figura 1. *Operatore And.*

x	y	x or y
Vera	Vera	Vera
Vera	Falsa	Vera
Falsa	Vera	Vera
Falsa	Falsa	Falsa

Figura 2. *Operatore Or.*

x	y	x nand y
Vera	Vera	Falsa
Falsa	Vera	Vera
Vera	Falsa	Vera
Falsa	Falsa	Vera

Figura 3. *Operatore Nand.*

Infatti, esattamente come prima, se le espressioni relazionali sono entrambe false, ad esse viene associato il valore 0 ed è pure 0 il valore della loro somma. In queste condizioni il salto condizionato non viene eseguito.

Se, viceversa, almeno una delle espressioni relazionali è vera, il computer vi associa il valore −1 e la somma risulta diversa da zero. In questo caso il salto condizionato viene eseguito.

Anche in questo caso non esiste alcun limite al numero di espressioni relazionali che possono venire legate da un operatore.

Per quanto riguarda l'operatore Not, il discorso è lievemente più delicato. Cercherò di chiarire con un esempio i problemi che si presentano.

Supponiamo di voler realizzare l'operatore logico Nand, definito mediante la figura 3. Ricordiamo che un Nand viene anche definito come un And seguito da un Not.

Basterà allora scrivere IF ((A>B) * (B<C) −1) THEN...

Infatti: se A>B e B<C sono entrambe vere, si ha ((A>B) * (B<C) −1) = (−1); * (−1) −1 = 1−1 = 0 e il salto condizionato non viene eseguito.

In ogni altro caso si ha 0 − 1 = −1 e il salto viene eseguito. Si vede che il comportamento complessivo è quello richiesto dall'operatore Nand.

Il problema è questo: (A>B) * (B<C) vale 1, se entrambe le espressioni relazionali sono vere, perché è il prodotto di un numero pari di espressioni relazionali vere, e cioè di un numero pari di −1.

Si capisce come il risultato sarebbe −1 se il numero delle espressioni relazionali fosse dispari.

In questo caso, al prodotto di un numero dispari di espressioni relazionali, non si deve sottrarre, ma bensì aggiungere 1. Quindi, se si vuole negare un And, si deve aggiungere al prodotto il numero 1, se le espressioni relazionali sono in numero dispari, mentre si deve sottrarre 1 se le espressioni relazionali sono in numero pari.

Oltre a ciò non c'è alcun problema a realizzare il Not.

Volendo negare un Or il discorso si fa più delicato. L'espressione (A>B) + (B<C) può assumere i valori 0, −1, −2. Inoltre il valore non può venire previsto a priori: dipende dai valori di A, B e C.

Per rimediare si può usare la funzione segno: SGN ((A>B) + (B<C)) assume i soli valori 0 e −1 a seconda che l'espressione A>B OR B<C sia falsa o vera.

A questo punto, per effettuarne la negazione, si procede come prima: IF 1 + SGN ((A>B) + (B<C)) THEN...

Concludo facendo notare come le tecniche illustrate permettano di estendere le applicazioni del computer verso il campo delle variabili Booleane.

Infatti è possibile, con le tecniche illustrate e con un po' di attenzione, definire variabili numeriche che assumano i valori 0 e −1 ed utilizzarle in espressioni complesse dell'algebra di Boole.

Ad esempio:

```
A = (B>C) * (C<>D) * (C=F);
B = (1 + SGN ((A>C) + (D=F)));
H = A*B
```

e così via.

Naturalmente, per creare delle espressioni coerenti, è necessario prestare molta attenzione agli stessi problemi che avevamo visto per il Not: bisogna che le variabili booleane siano sempre ricondotte ai valori base 0 e −1, per esempio, attraverso un attento uso della funzione segno.

DEBUG

Guida e cacciatore

Il signor Aurelio Fantone scrive per segnalare alcuni errori riscontrati nel programma "Guida e cacciatore" per ZX81 pubblicato su **Personal Software** *8-9.*

Alla linea 85 del listato 1 si legge:

```
LET X = Q
```

ovviamente, poiché sia garantita l'impossibilità di "sparare" due numeri uguali consecutivamente, la linea andrà coì corretta:

```
LET Q = X
```

Inoltre, nello stesso programma, occorrerà inserire una istruzione che impedisca di porre 0 (zero) quale "numero da raggiungere" (22 IF Z = 0 THEN GOTO 15). In caso contrario infatti, già alla prima "mossa" con un input (diviso) si conclude banalmente il gioco. Similmente, nel listato n. 2 la linea 5 andrebbe così corretta:

```
LET Z = 1 + INT (RND × 100)
```

Sempre nel listato n. 2 è presente un altro errore: dando per esempio come primo input 1 e come secondo "sparo" 2, sul video appare:

```
2 + 2 = 2
```

Ciò è dovuto al fatto che con simili input nessuna delle condizioni (linee 60-65-70-75) viene a realizzarsi.

Si pone rimedio a quanto sopra, senza alterare la logica del programma, variando la linea 60 nel modo seguente:

```
60 IF A< = B AND A<=C AND A<= D THEN
   GOTO 200
```

Sistemi ridotti per il Totocalcio per C64

Molti lettori ci hanno segnalato malfunzionamenti del programma, al quale, in effetti, come segnalatoci dall'autore Carlo Sintini, manca la seguente istruzione:

```
155 T$(1) = "1": T$(2) = "X": T$(3) = "2"
```

PICCOLI ANNUNCI

Varie

Vendo listati per tutti i computer compreso per BBC, Lynx, Oric 1, Atom, Dragon e tutte le altre marche più famose (Sinclair, Apple, Pet, ecc.). Inoltre vendo e cambio oltre 80 programmi per Spectrum. Iacopo Sannazzaro, Via Ginori 11, 50129 Firenze, tel. 287030.

Compro cassetta Jelli Monsters o scambio con una a scelta fra: Road Race, Jupiter Lander, Alien. Scambio anche programmi vari. Rispondo a tutti. Tommaso Gurrieri, Via Ugo Foscolo 14, 50124 Firenze, tel. 055/700635.

Vendo computer per scacchi Fidelity Electronics modello VSC, 10 livelli con infinito, scacchiera a sensori e led, orologio, 64 aperture, 64 grandi partite, parlante, legno pregiato. L. 650.000 (valore L. 950.000). G.C. Giacobbe, Via Finocchiara 46, 16144 Genova, tel. 010/825537.

Vendo TRS 80 livello 2 mod. 1 48 K, Ram 2 drive 5", monitor b/n 12", interfaccia Centronics, cavo per stampante e per 4 drive. Eccezionali programmi gestionali. Il tutto assemblato in elegante tavolo. Mario Bucolo, Via Sottomonte 5, 95030 Pedara (CT), tel. 095/915265.

Vendo Casio FX 702 P mai usato, ricevuto in regalo, garanzia, ottimo prezzo. Adriano Maggi, Via delle Rose 5, 20094 Corsico (MI), tel. 02/4405880.

Vendo ancora in garanzia monitor colore 20 pollici Sony KX 20 PS 1, L. 1.000.000 trattabili. Eccezionale espandibilità. Riceve Pal, Secam, RGB, NTFC. Fabrizio Barducci, Via Cicerone 1, 48015 Pinarella di Cervia (RA), tel. 988093.

Vendo Philips P 2000 T, 16 K RAM, CPU Z80, microcassetta digitale, colore, suono, interfaccia UHF, monitor floppy disk, RS 232. Completo di manuali e programmi sviluppati, il tutto a L. 1.000.000. Giuliano Franzosi, Via Zuretti 47, 20125 Milano, tel. 02/6083053.

Vendo listati per TRS 80 Atari 200/800 VIC 20, Apple II, Texas TI 99, Spectrum ZX 81, Pet. Mandare L. 1.000 per listino. Cambio anche programmi in cassetta per VIC 20. Contattare Massimo Gusso, Viale Felissent 32, 31020 Lancenigo (TV), tel. 0422/62469.

Vendo "Programmazione del 6502" edito Gruppo Editoriale Jackson. Prezzo copertina L. 22.000. Lo cedo per L. 16.000. Il libro è nuovo. Stefano Neri, Via Rosolina 157, 00010 Villa Adriana (Roma), tel. 0774/530792.

Vendo Casio PB 100 + interfaccia a cassette + 1 cassetta da 90 minuti con alcuni giochi. Il tutto (usato pochissimo) a L. 200.000. Paolo Bernardi, Via Brandolini d'Adda 20, 31100 Treviso, tel. 50729.

Utente Sirius *offre* scambio esperienze e programmi. Salvo Giudice, Via San Pio X 50, 31031 Caerano S. Marco (TV), tel. 0423/858201.

Vendo per Digital Rainbow 100 Multiplan con manuale L. 200.000; MBasic (8 bit) L. 150.000; Basic 86 (16 bit) L. 150.000. Silvia Barzaghi, Via Parini 42, 21047 Saronno (VA).

Per computer Atari 400-800 *vendo* i seguenti programmi: Pole Position, Qix, Flipper, Soccer, Digdug, Donkey Kong e molti altri su disco o cassetta. Inoltre vendo espansione di memoria da 48 K. Luigi Servolini, Via Simone de Saint Bon 25, 00195 Roma, tel. 06/384488-7581219.

Vendo videogame Atari con giochi Space Invaders, Video Olimpics, Miniature Golf, Bowling, Video Chess, Golf, Combat causa acquisto Commodore 64, il tutto a L. 399.000. Franco Francia, Via Cremosano 6, 20148 Milano, tel. 02/392084.

Vendo cambio programmi originali americani ed inglesi e in particolare giochi per il computer Atari 400/800 da L. 5.000 a 12.000. Scrivere o telefonare alla sera chiedendo di Andrea Vianello, Via S. Polo 2794, 30125 Venezia, tel. 041/36353.

Vendo TI 58 C Texas Instruments con biblioteca di base. Telefonare ora cena allo 02/718378. Marco Dal Monte, Via G.B. Nazari 3, 20129 Milano, tel. 718378.

Svendo Pac professionale e collaudato (nastro) L 373 scambi termici HP 85 + ROMS + STAMP 82905. Calcola K arch. sup. ponti term. verifica singoli locali intero edificio o *cambio* altri prog. professionali. Piero Pistoia, Via Mazzolari 2, 56045 Pomarance (Pisa).

Compro solo se letteralmente in pezzi o non più riparabile: HP 4I, stampante per HP 4I, HP 85. Vengo personalmente a ritirarlo. Pago in contanti. Telefonare dopo le 19. Emilio Guarise, Via Lessona 6, 20157 Milano, tel. 02/3570014.

Vendo HP 75 C luglio '83 in condizioni perfette, completo di mascherine, manuali, batterie, alimentatore, schede, ecc. Prezzo trattabile listino. Telefonare ore 20-21 Corrado, 06/6170169.

Compro TS Serie 800 se occasione o macchina simile (64 K, 0,5 Mb x 2), perfettamente funzionante e in zona. Gianni Fazzino, Via Manara 4, 21052 Busto Arsizio, tel. 0331/638511.

Vendo programmi verifica strutture in muratura metodo For in Basic per Commodore 64, Apple II e Uro 48 K, garanzia 3 mesi; 150 Setti per N piani, tutte le regioni. Mario Gardano, Viale Libia 209, 00199 Roma, tel. 836459.

Vendo programmi e progetti su fotocopia per tutti i computer pubblicati su: Bit-Personal Software MC microcomputer - "32 programmi per Apple". Prezzo L. 2.000 + L. 500 per spese di spedizione. Diego Lagunas, Via Nicolò Mauro 24, 31100 Treviso, tel. 0422/22898.

Vendo General Processor mod. T con 48 K RAM e dischi 8" 1024 Kbytes più stampante Epson MX 100. Vendo 50 dischi 8" a L. 4.000 l'uno con sopra molti programmi e linguaggi (Basic, Cobol, Fortran). Graziano Ceccotti, Via Livornese est 124, 56030 Perignano (PI), tel. 0587/616046.

Apple

Cambio-vendo programmi per Apple II e Apple //e di ogni tipo. Gestionali, grafica, ingegneria, game, The Last One con manuale in italiano, ecc. Giorgio Negrini, Via G. Pascoli 21, 46030 Cerese (MN), tel. 448131.

Vendo o cambio per Apple 2 programmi, Word Processing, Visicalc, Data Base, Utilitis, programmi di Copya (Locksmith, Nibbles Away, ecc.) e giochi (anche omaggio) a prezzi eccezionali!!! Andrea Giacomobono, Via Torbole 6, 00135 Roma, tel. 3201782.

Vendo a L. 20.000 scheda Controller Drive Standard Shugart per Apple, Drive a L. 350.000, scheda 128 K a L. 320.000, tastiera alfanumerica a L. 130.000. Roberto Pavesi, Viale Giulio Cesare 239, 28100 Novara, tel. 0321/454744.

Per Apple con CP/M 56-2.2 *vendo* Utility Tool per potenziare testo e grafica, inedito, possibilità eccezionali, compatibilità totale. Soft 80 col. HRCG Editor Hard Copy scambio montagne di software. Mirco Castellani, Via Ederle 7, 37023 Grezzana (VR), tel. 045/907998.

Cambio software per Apple II. Cerco inoltre persone disposte a formare un club di Apple in Sicilia. Mario Bucolo, Via Sottomonte 5, 95030 Pedara (CT), tel. 095/915265.

Compro per Apple II Language Card e/o scheda CPU Z80 in buone condizioni, anche se non originali Apple; inoltre compro o cambio con software fotocopie listato sorgente del DOS 3.3. Mario Bucolo, Via Sottomonte 5, 95030 Pedara (CT), tel. 095/915265.

Vendo scheda espansione 16 K RAM L. 100.000, scheda 80 colonne L. 120.000, scheda espansione 128 K RAM (DOS trasparente) L. 450.000. Tutto in ottimo stato per Apple II Plus. Isabella rag. Bottini, Via G. Galilei 681, 18038 Sanremo (IM).

Cambio-vendo programmi Apple. Inviate lista o telefonate ore serali. Tristano Marchini, Via Rosselli 6, 58033 Castel del Piano (GR), tel. 0564/955549.

Vendo programma calcolo e stampa modello 740 dichiarazione redditi completo di allegati ultra collaudato a L. 2.000.000. Funzionante su 4032/4040 8032/8050 e Apple o Apple II e 2 drive. D. Bettin, 35100 Padova, tel. 049/38678.

Cerco utenti di Apple II e da tutta Italia per scambio programmi. Franco Gasparroni, Via Siena 46, 63018 P. S. Elpidio (AP), tel. 994633.

Vendo Apple II Europlus 48 K con 4 manuali e imballo originali (+ 50 FTware omaggio) a L. 1.250.000. Usato pochissimo. Antonio Morsoletto, Via Marconi 85, 36077 Altavilla (VI), tel. 0444/981358.

Scambio software Apple II e cerco fotocopie del manuale del gioco Flight Simulator che posso pagare con un programma. Marco Carrubba, Via M. Campionesi 29, 20135 Milano, tel. 02/585294.

Cerco possessori di Apple Personal Computer per lavoro di battitura testi preferibilmente zona Milano. Giorgio Viappiani, Via Argonne 28, 20133 Milano, tel. 7384341.

Cerco persone disposte ad inviarmi programmi di qualsiasi genere per Apple II sebbene non possa ricambiare che con la mia riconoscenza la vostra generosità. Spese postali a mio carico. Maria Gualdi, Via Pegora 5, 44042 Cento (FE), tel. 051/906131.

Scambio corrispondenza con amici Apple II. Scambio anche listati. Inviate la vostra lista e io farò altrettanto. Per informazioni telefonare alle ore 10 al 23041 prefisso 0541 e chiedere di Max. Massimiliano Sartini, Via dei Mille 52, 47037 Rimini (FO), tel. 0541/23041.

Software Apple II Plus documentato ed utilizzabile *compro-cambio-vendo*. Esclusi giochi. Attivare contatto per liste particolareggiate. Anna Di Palma. Via Roma 124, 80030 Cimitile (NA)

Cambio-vendo programmi di ogni tipo per Apple II. Assicuro massima serietà. Cerco, inoltre, scheda Z80 da scambiare con equivalente in programmi. Michele Piscopo, P.za Marconi 9, 66013 Chieti, tel. 0871/582283.

Apple II, AIM 65 fotocopie *cerco* urgentemente: schemi elettrici, programmi Monitor e Applesoft Basic (Listing). Rimborso spese. Gianluca Sartori, Via Ungaretti 1, 31055 Quinto di Treviso (TV).

Vendo per Apple II 1 scheda Language Card L. 118.000; 1 SCH 80 col. L. 125.000; 1 SCH 80 col. (CP/M e Pascal compatibile) L. 250.000; 1 SCH interfaccia parallela tipo Centronics L. 75.000 ottimo stato. Rag. Isabella Bottini, Via G. Galilei 681, 18038 Sanremo (IM).

Vendo Language Card originale per Apple II e compatibili (prezzo di listino L. 352.000 + IVA) a L. 175.000. Vendo anche a mezzo postale. Tiziano Settimi, Via XXIV Maggio 30, 20010 Canegrate (MI), tel. 0331/400303.

Vendo software per Apple: Lock Smith, The Last One, Tasc Compiler, Paghe-Stip, Data Base Manag. Sist. Bollett. Fatt., RTTY (RX/TX), CW, Mailbox, Stazione radioamat., SSTV, Superscript. a prezzi d'elimin.ne. Stella Pietro, Via Visca 28, 67100 L'Aquila, tel. 23273.

Per Apple *vendo* scheda sintesi vocale per fonemi a L. 80.000. Luciano Bellotto, Via S. Pietro 10, 10034 Chivasso (TO), tel. 011/9111219.

Vendo Apple II Plus completo di potentissimi package di Word Processor, Data Bases, Professional, Teleprocessor, Grafica, Linguaggi, Compilatori, Commerciali, Ingegneria, Utility e numerosissimi giochi (Visifile, Visidex, Visicalc, Visiterm, CCA, DMS, PDB, Ptero, Prowriter, Graforth, Apple Plot,

COLECO VISION

Il piú Venduto negli Stati Uniti



Coleco Vision i nuovissimi Video Games, ad alta risoluzione grafica, pronti per Voi!

a casa vostra subito!

EXELCO

Via G. Verdi, 23/25
20095 - CUSANO MILANINO (MILANO)

Descrizione	Codice	Qt.	Prezzo unitario	Totale L.
Consolle con cartuccia Mouse Trap	68/8600-00		485.000	
Modulo convert. ATARI	68/8601-00		169.000	
Modulo Turbo	68/8602-00		130.000	
CARTUCCE SERIE ARGENTO				
Donkey Kong	68/8610-00		92.000	
Smurf (Puffi)	68/8610-01		92.000	
Zaxxon	68/8610-02		99.000	
Venture	68/8610-03		92.000	
Wizard of wor	68/8610-04		77.000	
Gorf	68/8610-05		77.000	
Mouse Trap	68/8610-06		77.000	
Carnival	68/8610-07		77.000	
Cosmic Avenger	68/8610-08		77.000	
Lady Bug	68/8610-09		77.000	

Desidero ricevere il materiale indicato nella tabella, a mezzo pacco raccomandato, contro assegno, al seguente indirizzo:

Nome

Cognome

Via

Città

Data C.A.P.

Partita I.V.A. o, per i privati Codice Fiscale

Sarà data precedenza alle spedizioni, se assieme all'ordine verrà incluso un anticipo di almeno L. 10.000

I prezzi sono comprensivi di IVA. Aggiungere L. 5.000 per il recapito a domicilio.

BI/11-83

PICCOLI ANNUNCI PERSONAL SOFTWARE

Sei un lettore di PERSONAL-SOFTWARE e vuoi entrare in contatto con tutti gli altri lettori per comperare, cambiare o vendere software? Spedisci questo tagliando a Gruppo Editoriale Jackson - Via Rosellini, 12 - 20124 Milano

Nome .... Cognome ....

Via .... C.A.P. ....

Città .... Tel. ....

CEDOLA DI COMMISSIONE LIBRARIA PERSONAL SOFTWARE

Da inviare a Gruppo Editoriale Jackson - Via Rosellini, 12 - 20124 Milano

Nome Cognome

Indirizzo

Cap. Città Provincia

Partita I.V.A. (indispensabile per le aziende)

Si richiede l'emissione della fattura □

Inviatemi i seguenti libri:

Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità

□ Pagherò al postino il prezzo indicato + L. 2.000 per contributo fisso spese di spedizione.

□ Allego assegno n° .... di L. ....

□ Non abbonato □ Abbonato sconto 20% □ l'Elettronica □ Elettronica Oggi □ Automazione Oggi □ Elektor
□ Informatica Oggi □ Computerworld □ Bit □ Personal Software □ Strumenti Musicali □ Videogiochi

Data .... Firma ....

UNA PUBBLICAZIONE
DEL GRUPPO EDITORIALE JACKSON

PERSONAL SOFTWARE

ANNO 2 N.10-11 SETTEMBRE-OTTOBRE 1983

DIRETTORE RESPONSABILE: Giampietro Zanga
DIRETTORE: Pietro Dell'Orco
REDAZIONE: Lorenzo Barrile
COORDINAMENTO: Riccardo Paolillo
CONSULENZA: Giacomo Bortone
HANNO COLLABORATO A QUESTO NUMERO:
A. Chessa, G. Confortini, B. Del Medico, A. Guida, M. Morchio, M. Petraccone, F. Sardo, A. Zannoni

GRAFICA E IMPAGINAZIONE: Luigi Chiesa
PUBBLICITÀ: Concessionario per l'Italia e l'Estero
Reina s.r.l. Via Washington, 50 - 20146 Milano
Tel. (02) 4988066/7/8/9/060 (5 linee r.a.)
Telex 316213 REINA I
STAMPA: Reweba (Brescia)

Concessionario esclusivo per la DIFFUSIONE in Italia e all'Estero:
SODIP - Via Zuretti, 25 - 20125 Milano

AUTORIZZAZIONE ALLA PUBBLICAZIONE: Tribunale di Milano n. 69 del 20/2/1982

Spedizione in abbonamento Postale Gruppo III/70
Prezzo della rivista L. 3.500. Numero arretrato L. 6.000.
Abbonamento annuo (10 numeri) L. 35.000; per l'Estero L. 48.000
I versamenti vanno indirizzati a: Gruppo Editoriale Jackson - Via Rosellini, 12 - 20124 Milano - mediante emissione di assegno bancario, cartolina vaglia o utilizzando il c/c Postale numero 11666203.
Per i cambi di indirizzo, indicare, oltre naturalmente al nuovo, anche l'indirizzo precedente, ed allegare alla comunicazione l'importo di L. 500, anche in francobolli.

Il Gruppo Editoriale Jackson è iscritto nel registro Nazionale della Stampa al n. 117 - Vol 2. - Foglio 129 in data 17-8-1982

GRUPPO EDITORIALE JACKSON Srl

DIREZIONE, REDAZIONE, AMMINISTRAZIONE:
Via Rosellini, 12 - 20124 Milano - Telefoni: 68.03.68 - 68.00.54
SEDE LEGALE: Via Vincenzo Monti, 15 - 20123 Milano
DIREZIONE EDITORIALE: Giampietro Zanga e Paolo Reina
COORDINAMENTO EDITORIALE: Daniele Comboni
DIREZIONE AMMINISTRATIVA: Giuliano di Chiano

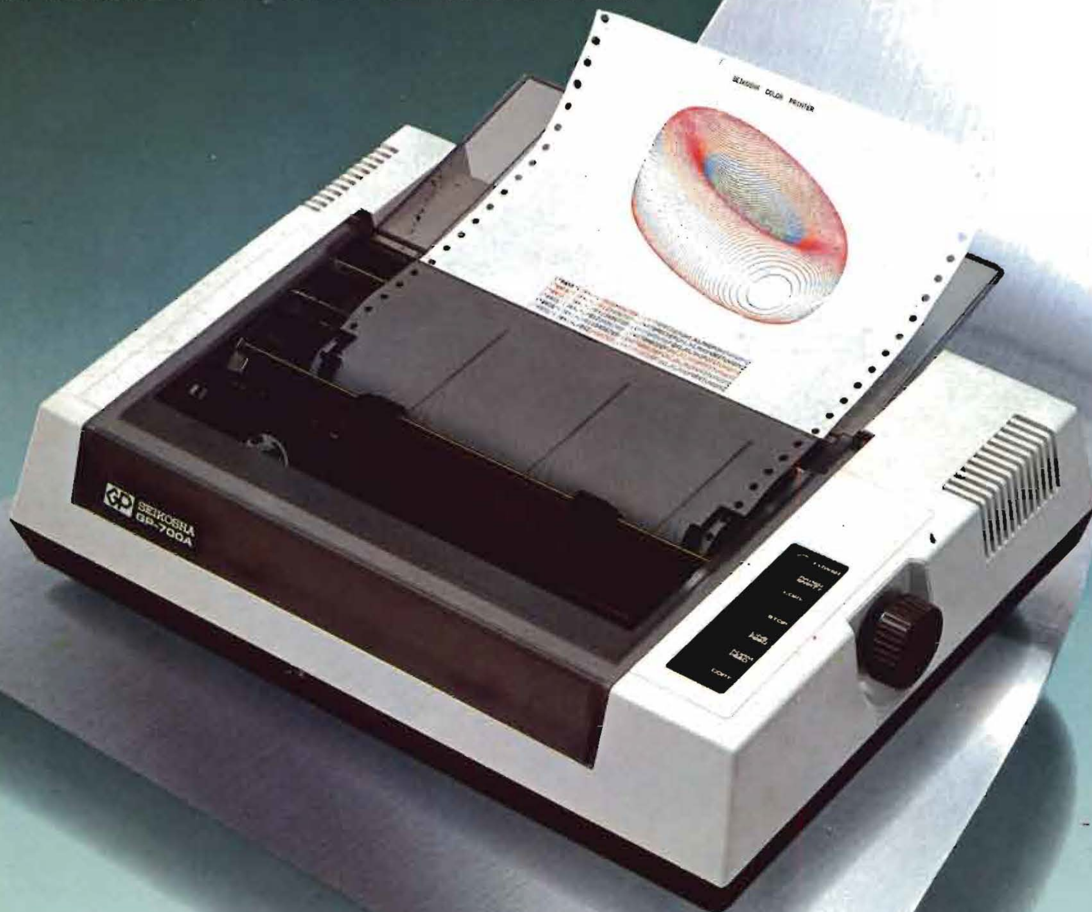
QUANTI COLORI HA LA TUA STAMPANTE ?

NEL 1983 LA SEIKOSHA PER PRIMA AL MONDO
E' IN GRADO DI PRESENTARE LA NUOVA STAMPANTE
GRAFICA A SETTE COLORI.

RIUNITE IN UN APPARECCHIO PRATICO E COMPATTO
LE CARATTERISTICHE DELLA STAMPANTE E DEL PLOTTER,
LA SEIKOSHA INVENTA UN NUOVO TIPO DI PERIFERICA
CHE BEN PRESTO SARA' INSOSTITUIBILE.

REBIT COMPUTER E' ORGOGLIOSA DI LANCIARE
QUESTA NOVITA' ASSOLUTA SUL MERCATO ITALIANO
AD UN PREZZO MOLTO, MOLTO COMPETITIVO:
MENO DI UN MILIONE.
MENO DI UNA COMUNE STAMPANTE IN BIANCONERO.

REBIT COMPUTER
A DIVISION OF G.B.C.



GP-700A

Graphic Color Printer

SEIKOSHA

REBIT COMPUTER - Divisione della GBC Italiana S.p.A. -
20092 CINISELLO BALSAMO - Tlx 330028 GBCMIL - Casella Postale 10488 MI

IL BASIC E LA GESTIONE DEI FILE

Il libro si rivolge in modo particolare a chi già conosce il Basic e desidera poter realizzare programmi che prevedano l'uso di file residenti su disco. Dopo aver preso in esame, utilizzando numerosi esempi pratici, le particolarità del Microsoft, si passa alla descrizione delle istruzioni necessarie ad una corretta gestione dei file su disco, sia ad accesso diretto che sequenziale. Una terza parte del libro è infine interamente dedicata alla esposizione dei metodi pratici per l'uso dei file ad accesso diretto e dei data base.

Cod. 515H **L. 11.000 Pagg. 164**

50 ESERCIZI IN BASIC

Una raccolta completa e progressiva di esercizi riguardanti matematica, gestione, ricerca operativa, gioco e statistica. Ciascun esercizio proposto comporta l'enunciazione e l'analisi del problema, la risoluzione mediante flow-chart e commenti, così come un programma che implementa la soluzione, illustrato da semplici esempi rappresentativi.

Questo metodo mette in grado il lettore di verificare passo passo le sue coñoscenze e il livello di apprendimento raggiunto.

Cod. 521A **L. 13.000 Pagg. 208**

75 PROGRAMMI IN BASIC PER IL VOSTRO COMPUTER

Il volume raccoglie ben 75 programmi originali scritti in un Basic generico, utilizzabili, salvo poche e semplici modifiche, sulla maggior parte dei personal computer in commercio, a cassetta come a floppy disk. Per ciascuno, dopo una descrizione introduttiva, viene fornito il listing e un campione di esecuzione. Così come sono, i programmi proposti (tutti verificati), costituiscono un valido ausilio per chiunque debba risolvere problemi di matematica, statistica, finanza o, genericamente, di pratica utilità.

Cod. 551D **L. 12.000 Pagg. 196**

GIOCARE IN BASIC

Nei giochi, il lettore può ritrovare tutte quelle situazioni reali di programmazione che gli saranno indispensabili nella comprensione e realizzazione di qualsiasi applicazione interattiva del proprio computer, anche le più sofisticate. Questo senza annoiarsi, ma entrando da subito all'interno della materia per imparare a comprendere il BASIC, il proprio computer e i computer in genere.

Cod. 522A **L. 20.000 Pagg. 324**

... dalla libreria JACKSON



DIVISIONE LIBRI

CEDOLA DI COMMISSIONE LIBRARIA

Ritagliare (o fotocopiare) e inviare a
Gruppo Editoriale Jackson
Via Rosellini, 12 - 20124 Milano

Nome e Cognome ______________________

____

Indirizzo ______________________

____

Cap. __________ Città ______________________

Provincia ______________

Codice Fiscale (indispensabile per le aziende)

Inviatemi i seguenti libri:

Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità

□ Pagherò al postino il prezzo indicato + L. 2.000 per contributo fisso spese di spedizione

□ Allego assegno n° .... di L. ....

Data .... Firma ....